

Lecture 04

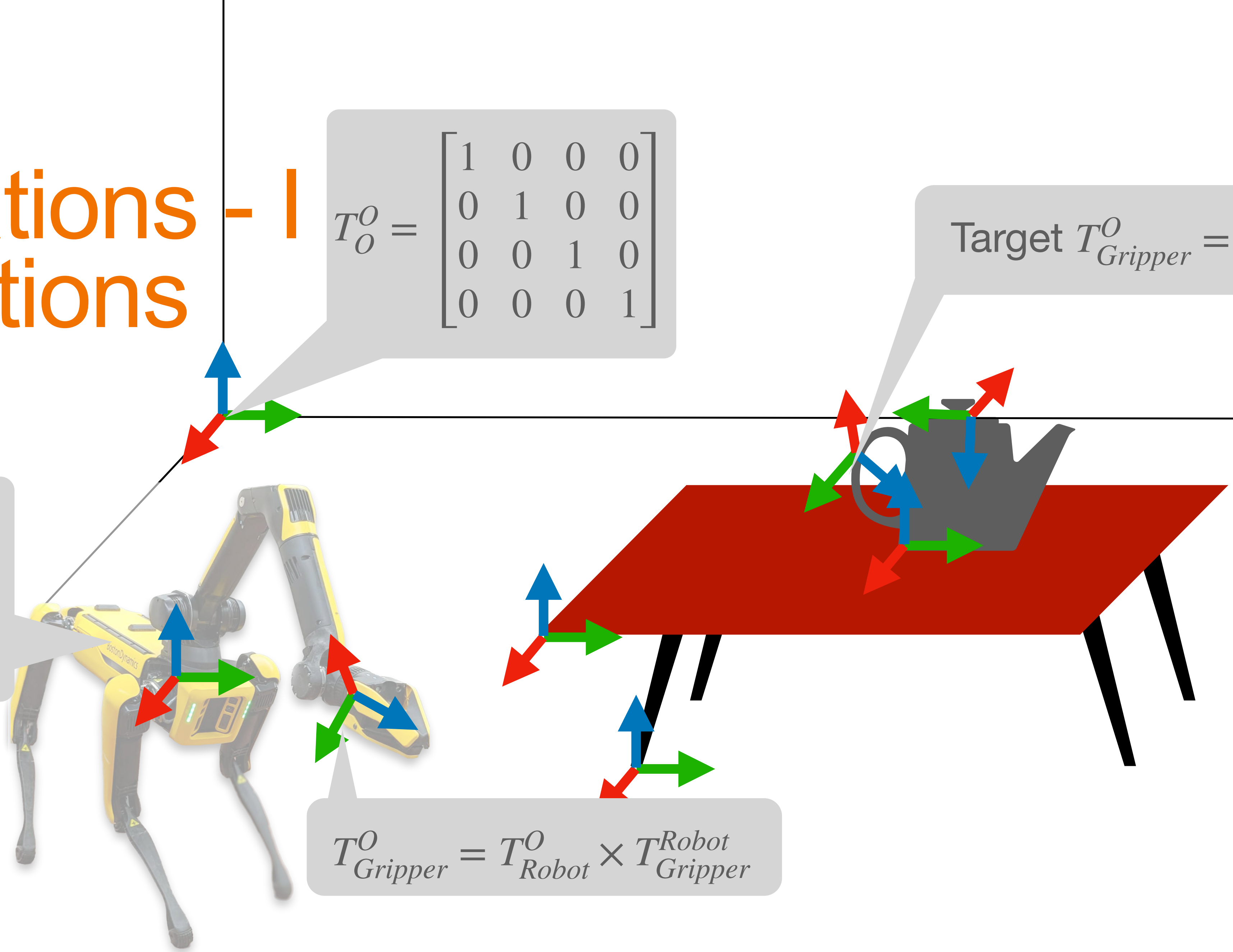
Representations Transformations

$$T_O^O = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Target $T_{Gripper}^O = T_{Jar}^O$

$$T_{Robot}^O = \begin{bmatrix} R_{3 \times 3} & D_{3 \times 1} \\ 0_{1 \times 3} & 1 \end{bmatrix} \times \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$$T_{Gripper}^O = T_{Robot}^O \times T_{Gripper}^{Robot}$$



Course Logistics

- Everyone should be on Ed discussion board now.
- Everyone should be on Gradescope now.
- Quiz 2 will be posted tomorrow (Tuesday) 6pm and will be due on 02/05 (Wednesday) **noon**.
- Project 1 is due on **02/05 (Wednesday) 11:59 pm CT**.
- Project 2 will be released on 02/05.
- Autograder is available. Please check to see if you have access to it. Recently enrolled students please check this.
 - 10 submissions per day. This is a good coding practice and we will not increase it.
 - You don't need a valid solution to test autograder. So if you haven't submitted anything just try it first.



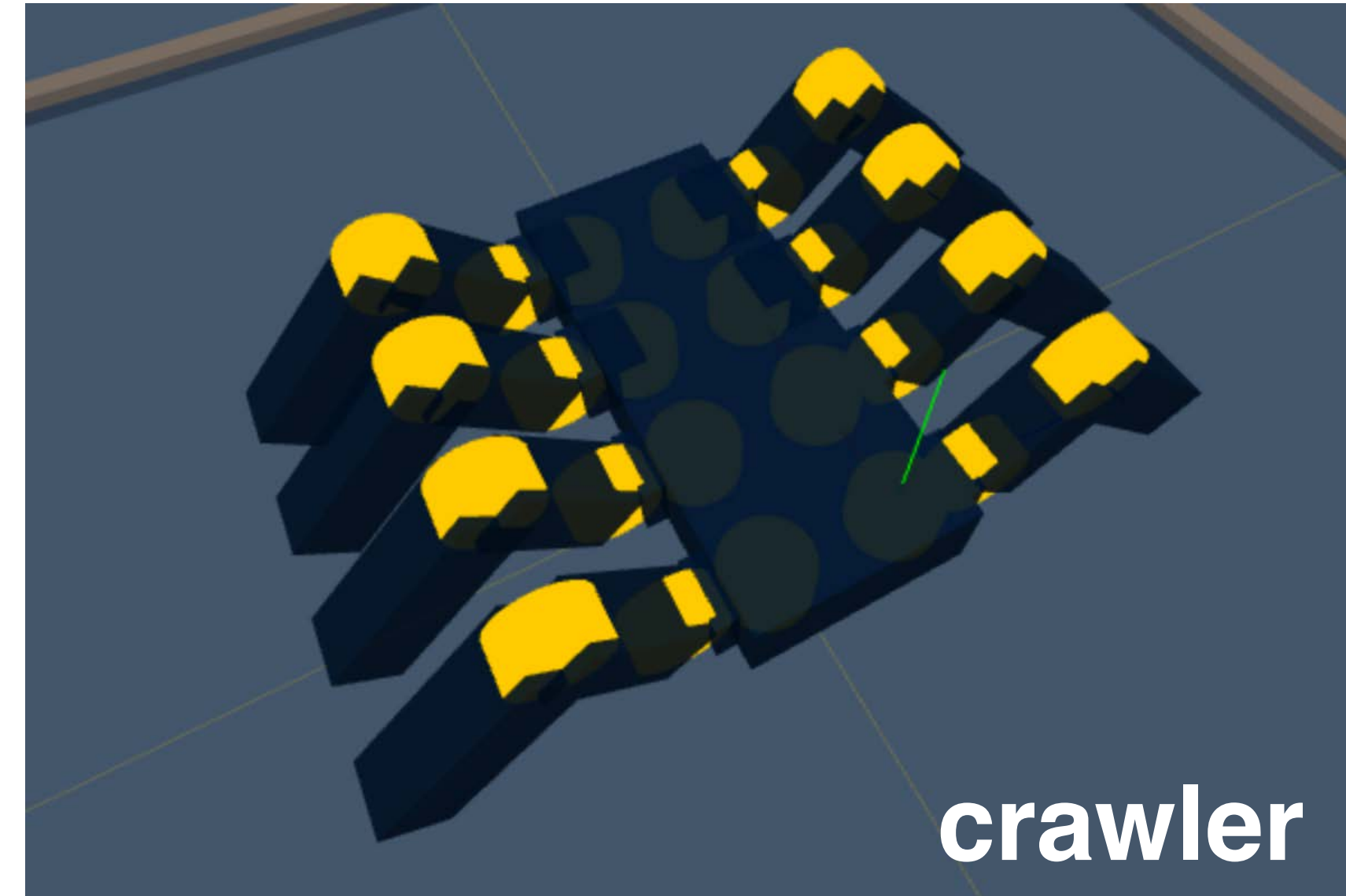
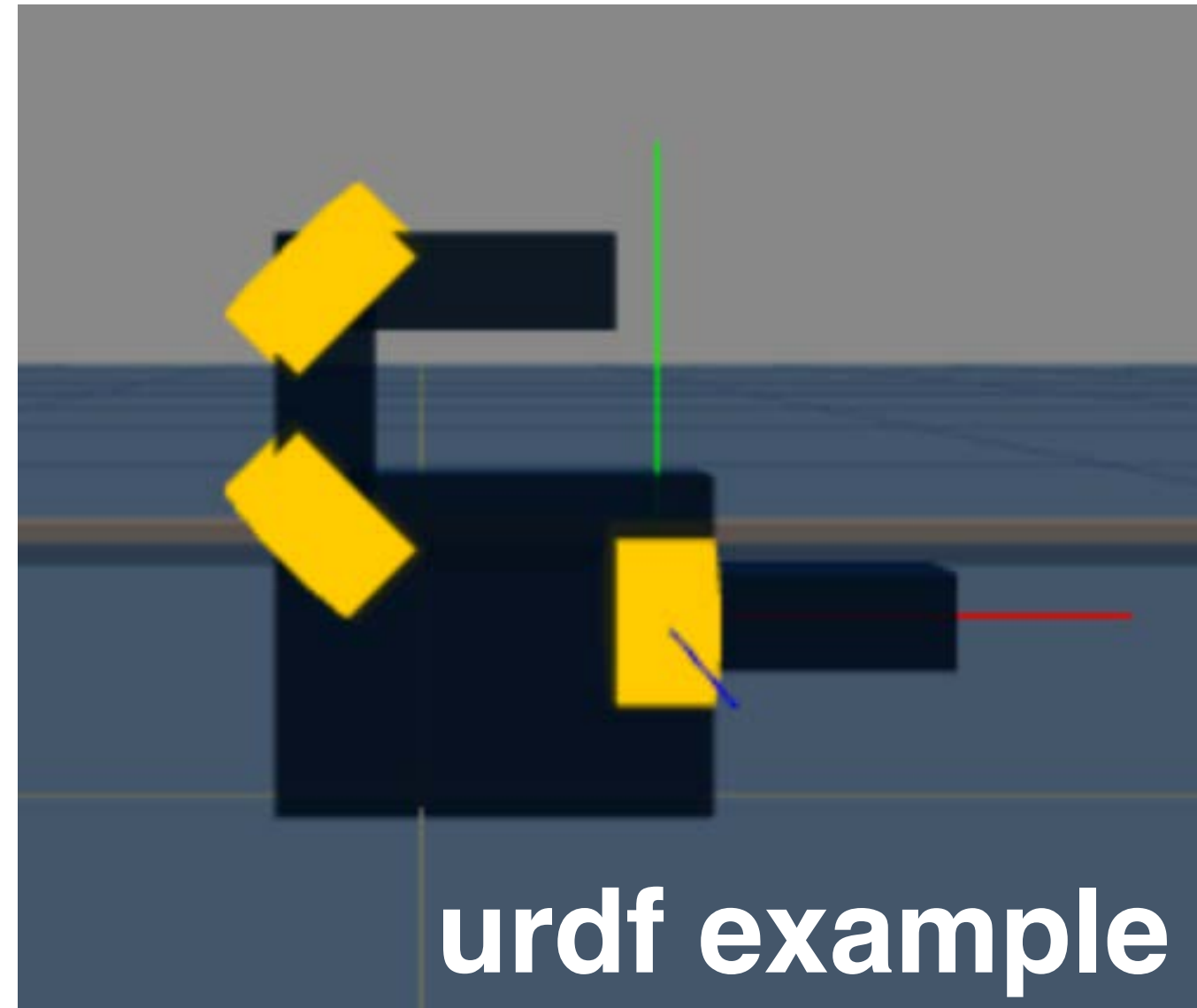
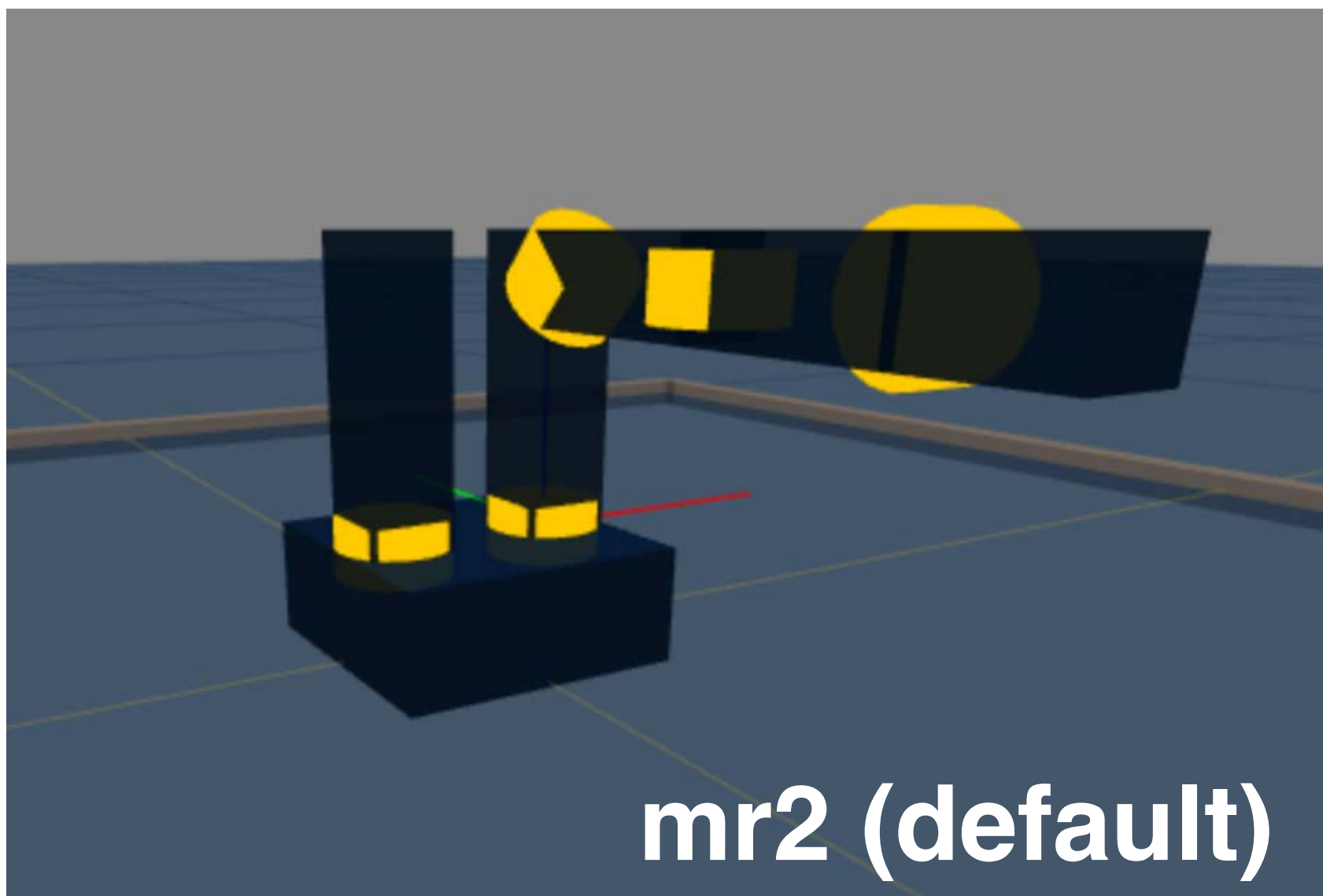
Course Logistics

What does 3 **total** late day tokens mean for P1-6?

- 3 total late day tokens are for
 - If you fell sick and wanted an extra day or two to submit a project.
 - If you had to travel and wanted an extra day or two to submit a project.
 - If you got overwhelmed with other course projects and wanted a day or two to submit.
- Example:
 - If you used 2 late day tokens for your Project-1, then you are left with 1 late day tokens for Project 2-6.
- So save up your late tokens for later projects in the course, unless you need it for the earlier ones.

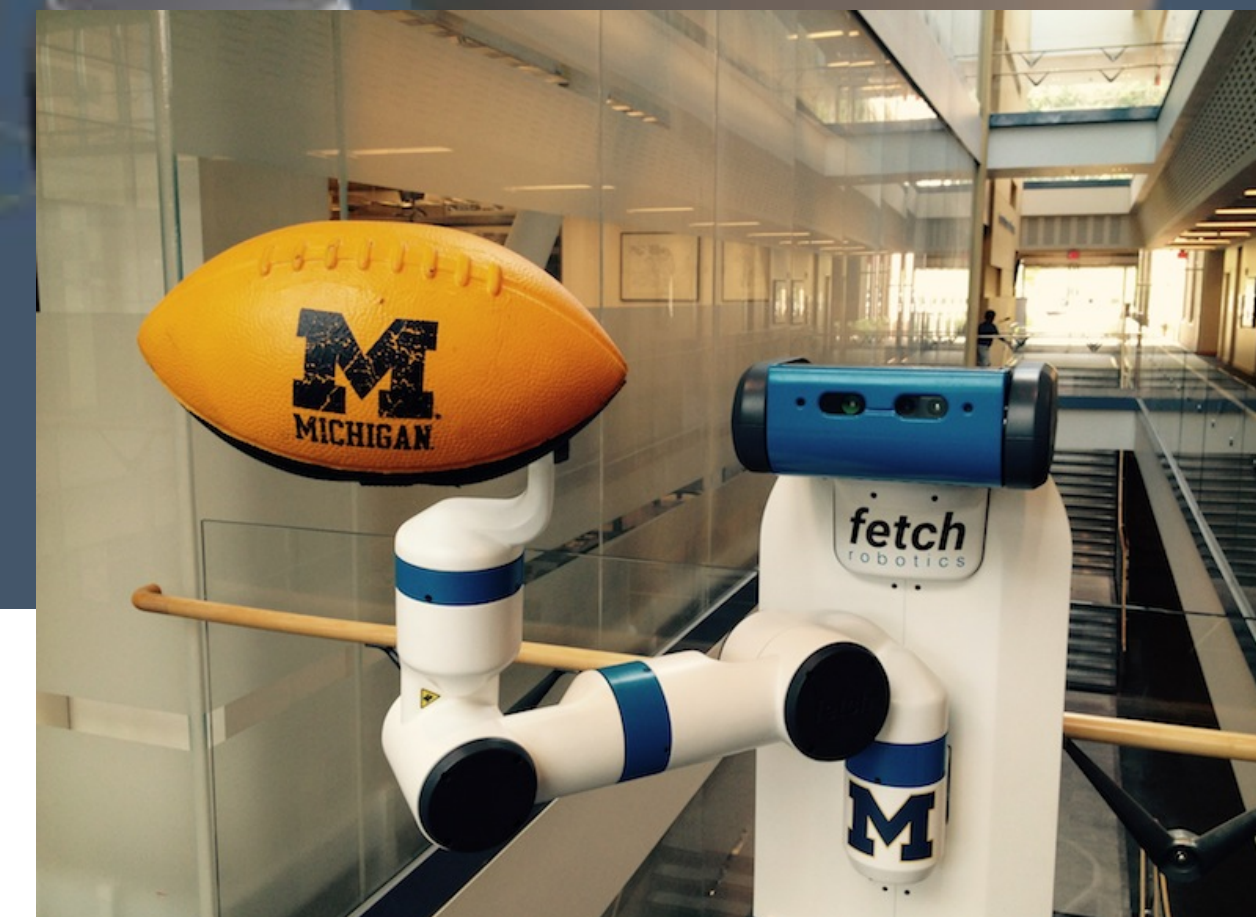
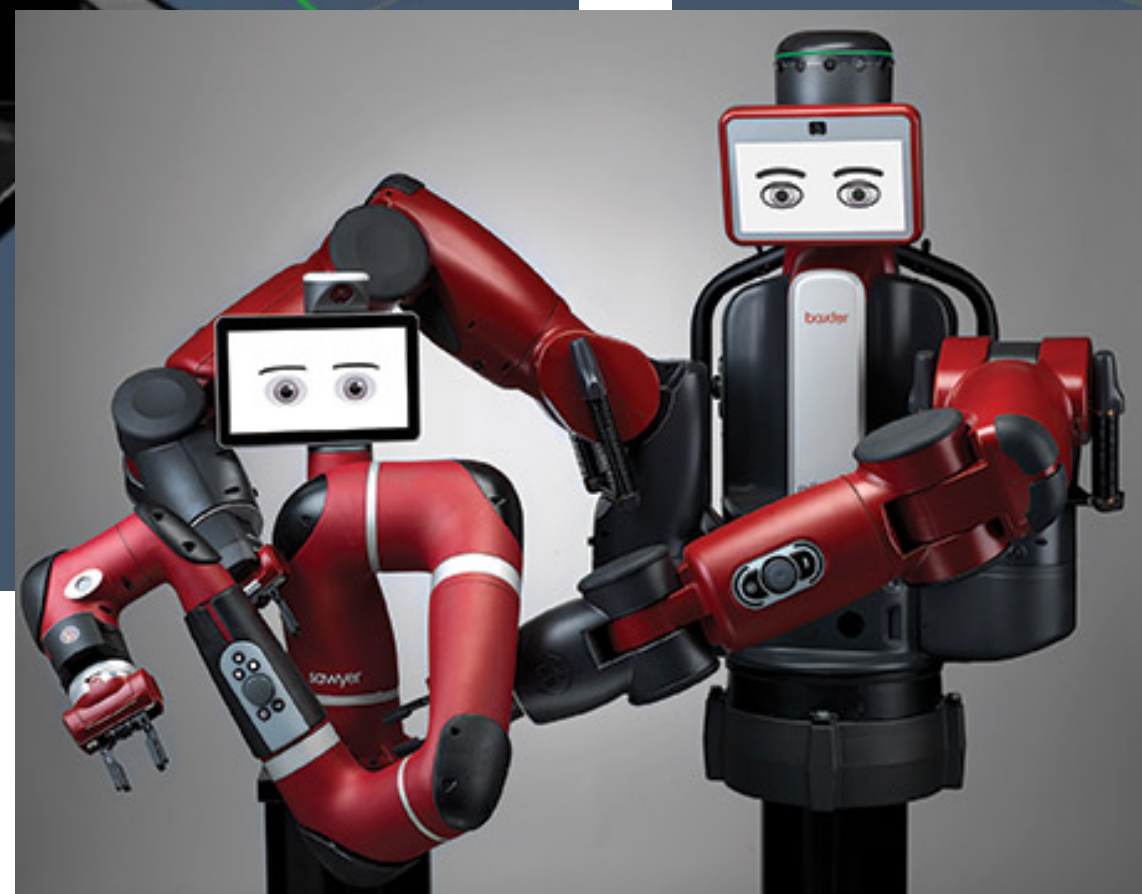
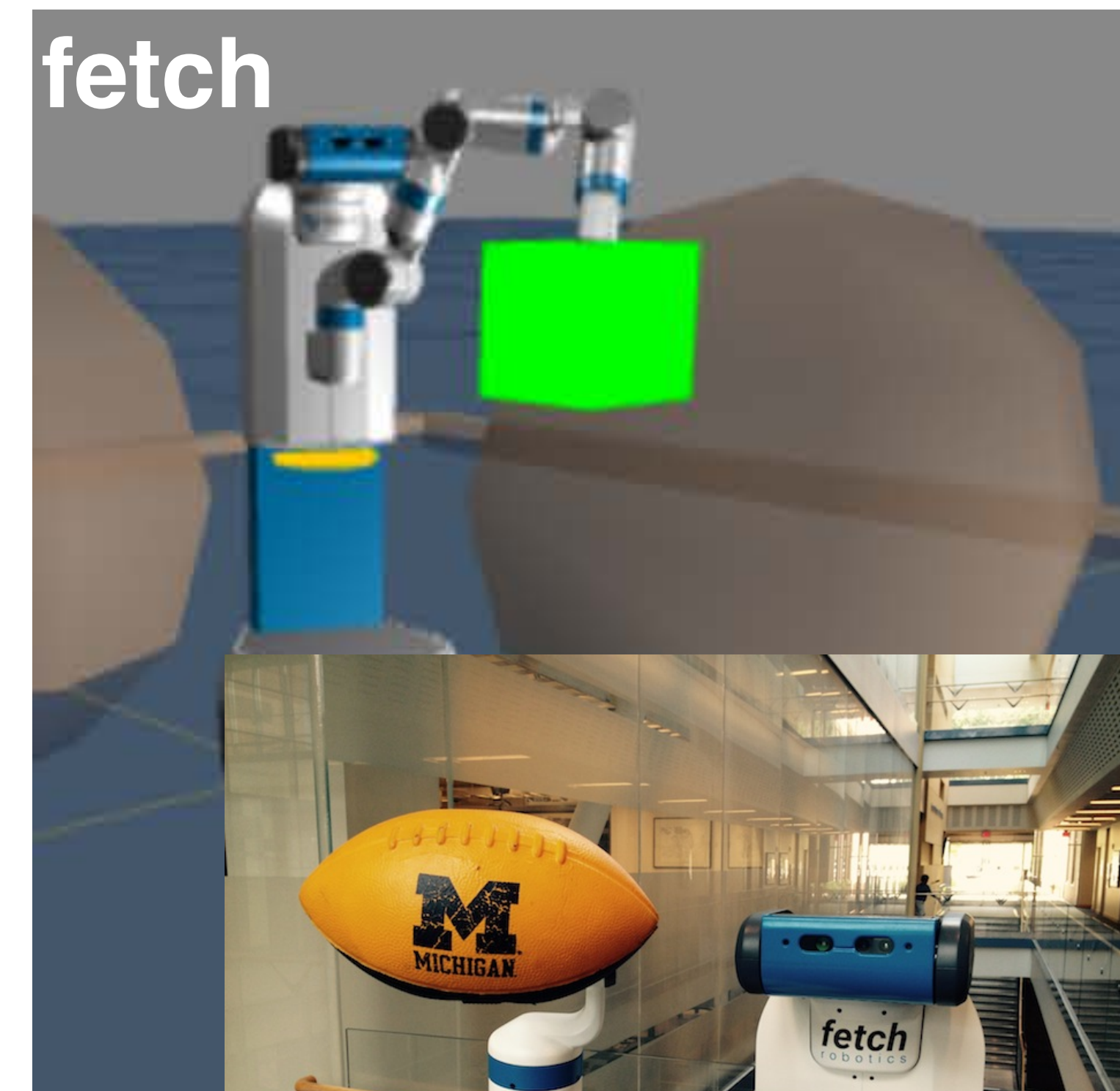
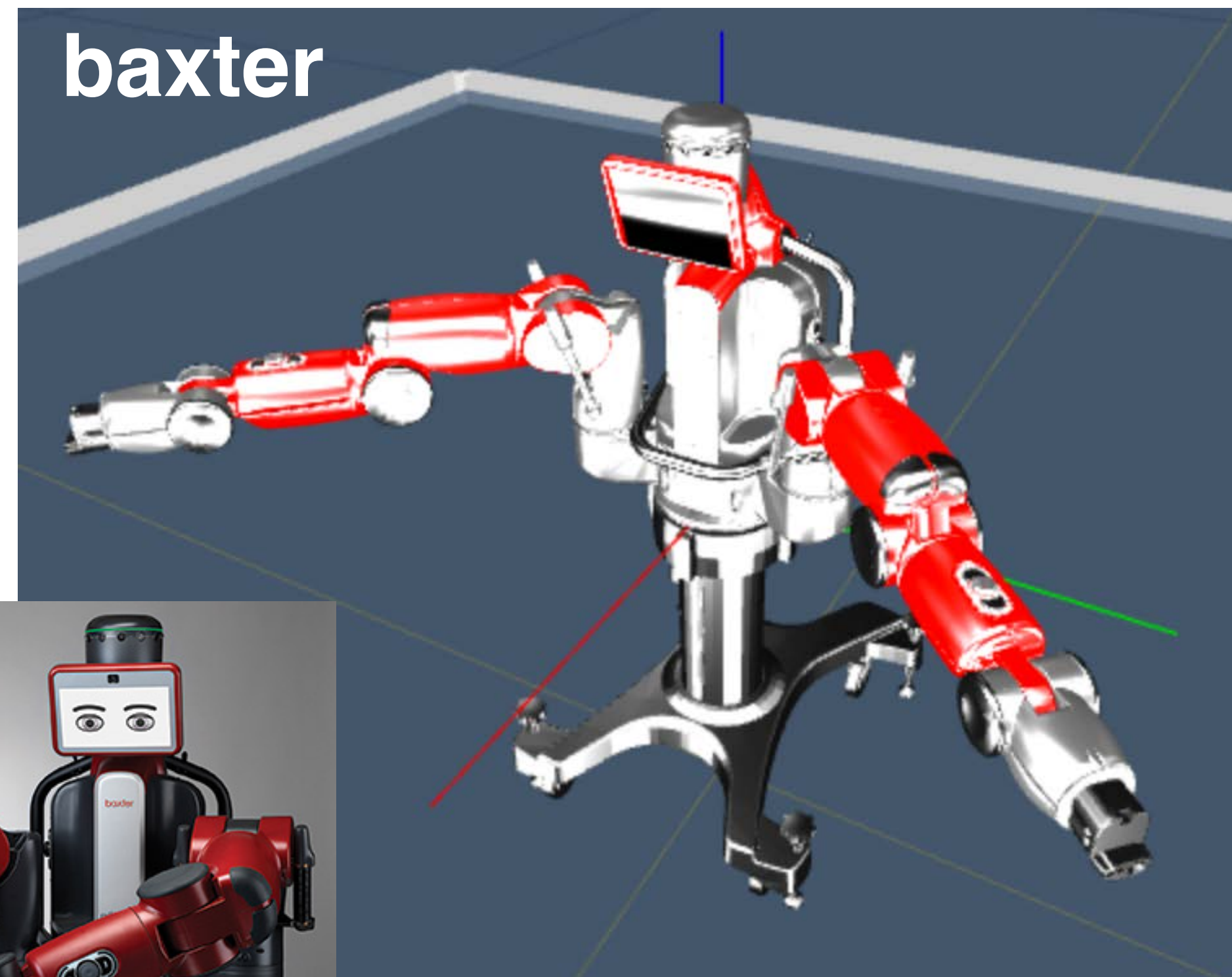
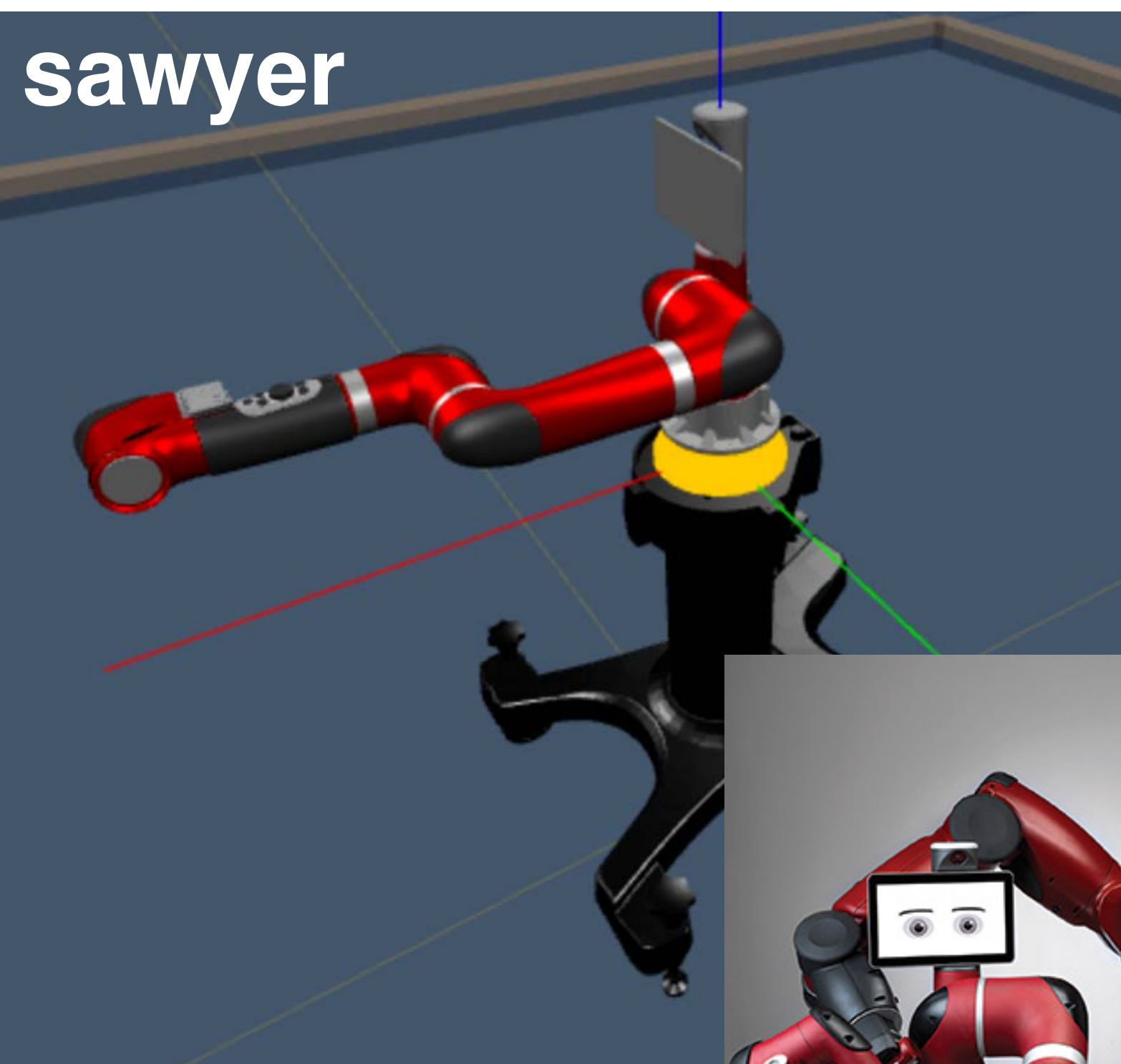


Why JavaScript?



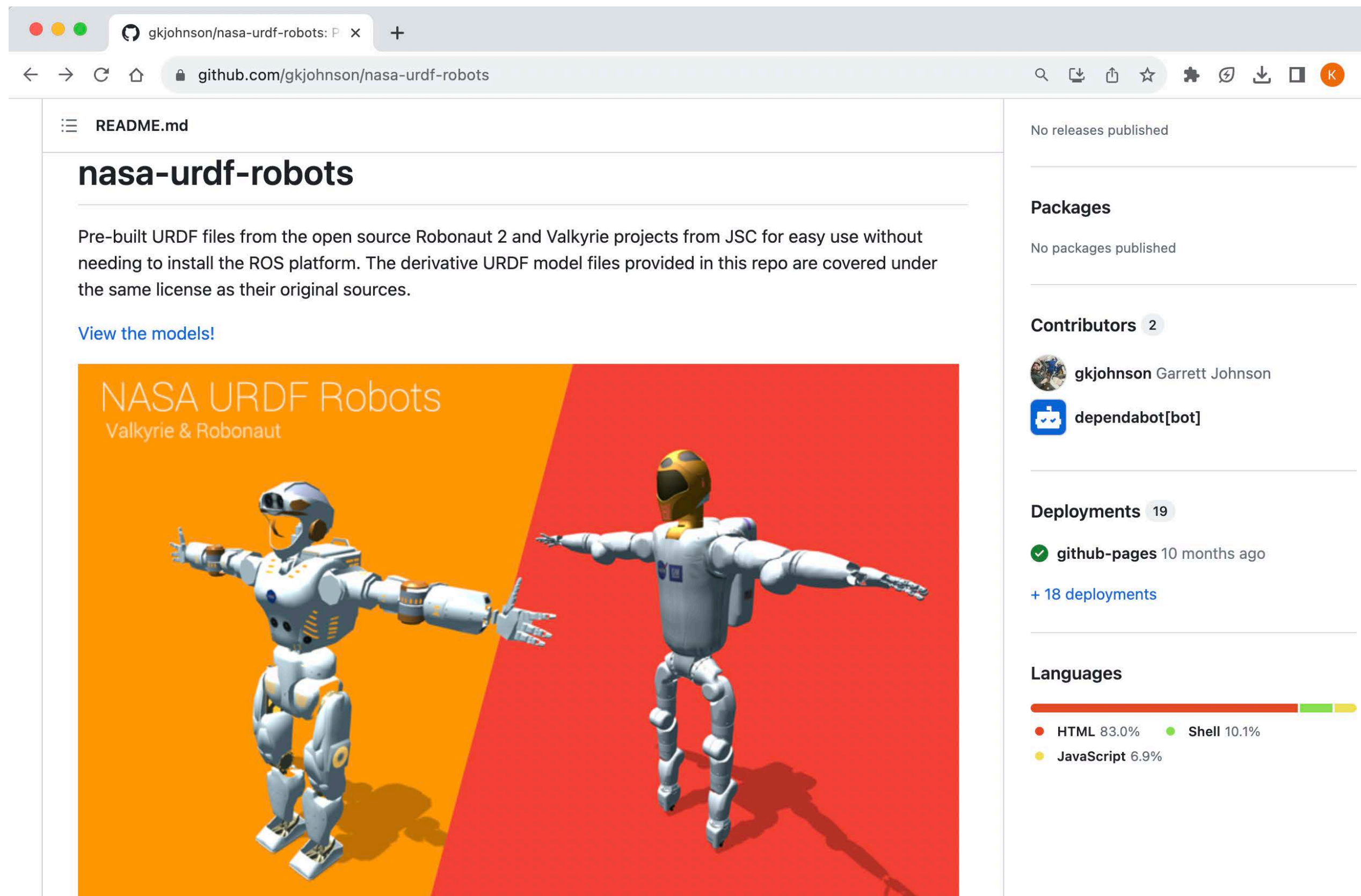
Why JavaScript?

You can load a URDF of a famous robot!



Why JavaScript?

More robot models!!!



The screenshot shows a web browser displaying the GitHub repository for 'nasa-urdf-robots'. The page title is 'nasa-urdf-robots' and the description reads: 'Pre-built URDF files from the open source Robonaut 2 and Valkyrie projects from JSC for easy use without needing to install the ROS platform. The derivative URDF model files provided in this repo are covered under the same license as their original sources.' Below the description is a link 'View the models!'. A large image shows two humanoid robots, one on an orange background and one on a red background, with the text 'NASA URDF Robots Valkyrie & Robonaut'. To the right of the main content, there are statistics: 'No releases published', 'No packages published', 'Contributors 2' (listing gkjohnson and dependabot[bot]), 'Deployments 19' (listing github-pages), and a 'Languages' bar chart showing HTML (83.0%), Shell (10.1%), and JavaScript (6.9%).



You can try to load URDFs of our lab robots too!
Ask the course staff for the URDFs.

Previously

Reset: DOFs and Coordinate Spaces

- Each body has its own frame

Rigid Body
vs.
Link
vs.
Joint

- Spatial geometry attached to each link, but does not affect the body's coordinate frame

frames, hinge, prismatic, ball

2D Example

any point on the line satisfies
only single point satisfies both lines
no point satisfies all three lines

One equation, Two equations, Three equations

Vector Addition and Subtraction

$$a + b = \begin{bmatrix} a_x + b_x \\ a_y + b_y \\ a_z + b_z \end{bmatrix}$$

vector addition visually

vector addition is order independent

vector subtraction is addition with negated vector

Magnitude and Unit Vector

The magnitude of a vector is the square root of the sum of squares of its components

$$\|a\| = \sqrt{a_1^2 + a_2^2 + \dots + a_n^2}$$

A unit vector has a magnitude of one. Normalization scales a vector to unit length.

$$\hat{a} = \frac{a}{\|a\|}$$

A vector can be multiplied by a scalar

$$sa = \begin{bmatrix} sa_x \\ sa_y \\ sa_z \end{bmatrix}$$

Dot Product

scalar result

$$a \bullet b = a_x b_x + a_y b_y + a_z b_z = \|a\| \|b\| \cos(\theta)$$

Measures the similarity in direction of two vectors

$$\begin{bmatrix} 2 \\ 1 \end{bmatrix} \cdot \begin{bmatrix} 3 \\ 2 \end{bmatrix} = 2 * 3 + 1 * 2 = 8$$

Projections

Dot products related to projections onto vectors.

Scalar projection of one vector onto another

$$a_1 = |a| \cos \theta = a \cdot \hat{b} = a \cdot \frac{b}{|b|}$$

Vector projection

$$a_1 = a_1 \hat{b}$$

\hat{b} is unit length

Cross Product

Assumes **a** and **b** are in same frame

$$c_x = a_y b_z - a_z b_y$$

$$c_y = a_z b_x - a_x b_z$$

$$c_z = a_x b_y - a_y b_x$$

Results in new vector **c** orthogonal to both original vectors **a** and **b**

Length of vector **c** is equal to area of parallelogram formed by **a** and **b**

$$\|a \times b\| = \|a\| \|b\| \sin \theta$$

Matrix-vector multiplication

(two interpretations)

1) **Row story:** dot product of each matrix row

$$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} j \\ k \\ l \end{bmatrix} = \begin{bmatrix} aj + bk + cl \\ dj + ek + fl \\ gj + hk + il \end{bmatrix}$$

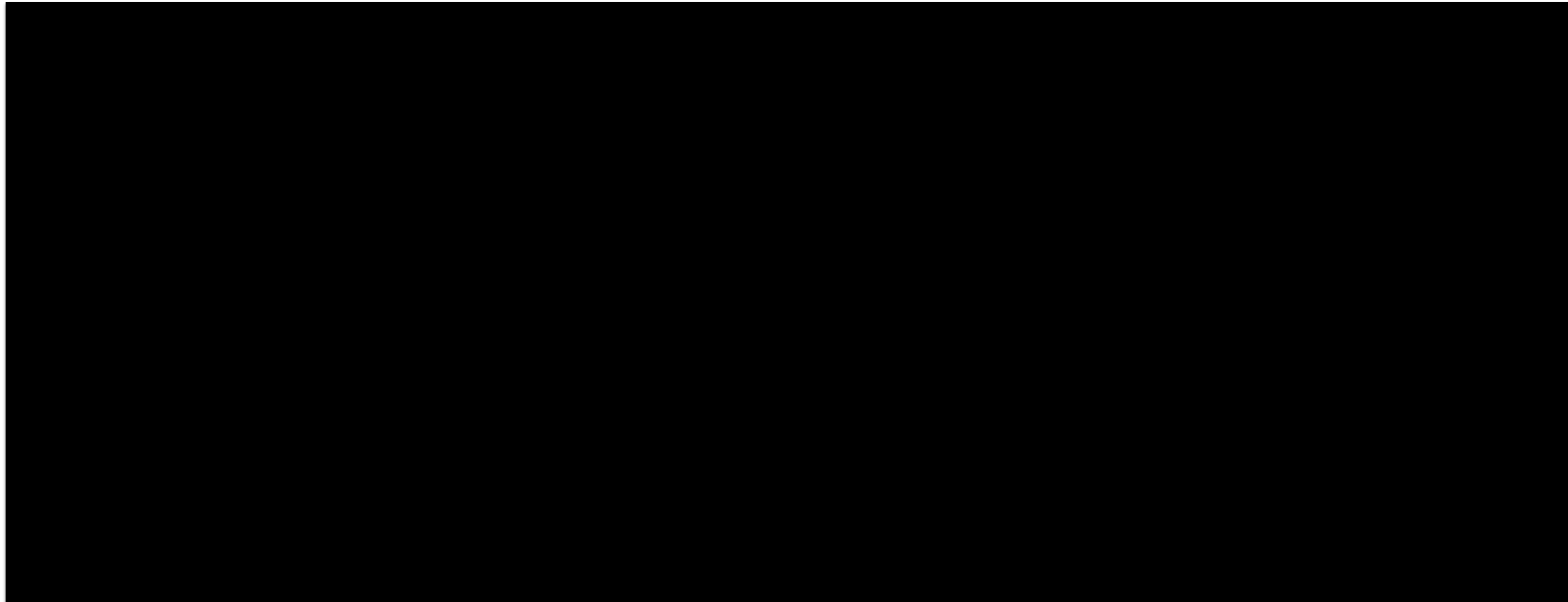
2) **Column story:** linear combination of matrix columns

$$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} j \\ k \\ l \end{bmatrix} = \begin{bmatrix} aj + bk + cl \\ dj + ek + fl \\ gj + hk + il \end{bmatrix} = \begin{bmatrix} a \\ d \\ g \end{bmatrix} j + \begin{bmatrix} b \\ e \\ h \end{bmatrix} k + \begin{bmatrix} c \\ f \\ i \end{bmatrix} l$$

Solving linear systems

What would be the direct way to solve for \mathbf{x} ?

$$A\mathbf{x} = \mathbf{b}$$



Solving linear systems

What would be the direct way to solve for \mathbf{x} ?

$$\mathbf{Ax} = \mathbf{b}$$

Invert \mathbf{A} and multiply by \mathbf{b}

$$\mathbf{x} = \mathbf{A}^{-1}\mathbf{b}$$



Matrix rank and inversion

- Let A be a square n by n matrix. A is invertible if full rank and a matrix B exists such that
- Rank of a matrix A is the size of the largest collection of linearly independent columns of A
- A is invertible (nonsingular) if it has full rank
- Gaussian elimination can find matrix inverse
- Singular matrix cannot be inverted this way

$$\mathbf{AB} = \mathbf{BA} = \mathbf{I}_n$$

$$[A|I] = \left[\begin{array}{ccc|ccc} 2 & -1 & 0 & 1 & 0 & 0 \\ -1 & 2 & -1 & 0 & 1 & 0 \\ 0 & -1 & 2 & 0 & 0 & 1 \end{array} \right]$$

$$[I|B] = \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & \frac{3}{4} & \frac{1}{2} & \frac{1}{4} \\ 0 & 1 & 0 & \frac{1}{2} & 1 & \frac{1}{2} \\ 0 & 0 & 1 & \frac{1}{4} & \frac{1}{2} & \frac{3}{4} \end{array} \right]$$



Solving linear systems

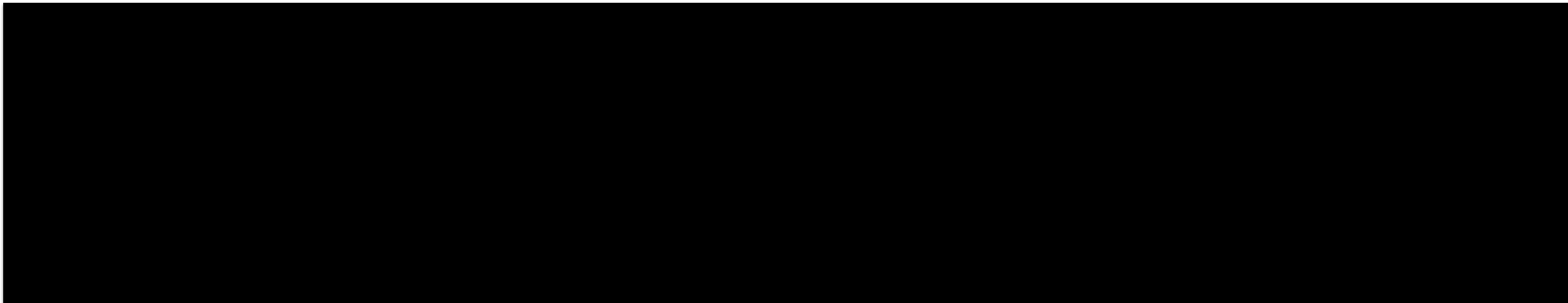
What would be the direct way to solve for \mathbf{x} ?

$$\mathbf{Ax} = \mathbf{b}$$

Invert \mathbf{A} and multiply by \mathbf{b}

$$\mathbf{x} = \mathbf{A}^{-1}\mathbf{b}$$

Can this always be done?



Solving linear systems

What would be the direct way to solve for \mathbf{x} ?

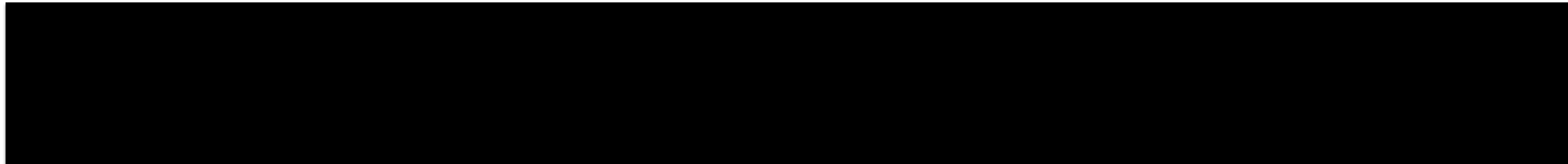
$$\mathbf{Ax} = \mathbf{b}$$

Invert \mathbf{A} and multiply by \mathbf{b}

$$\mathbf{x} = \mathbf{A}^{-1}\mathbf{b}$$

Can this always be done?

No. But, we can approximate. How?



Solving linear systems

What would be the direct way to solve for \mathbf{x} ?

$$\mathbf{Ax} = \mathbf{b}$$

Invert \mathbf{A} and multiply by \mathbf{b}

$$\mathbf{x} = \mathbf{A}^{-1}\mathbf{b}$$

Can this always be done?

No. But, we can approximate. How?

Pseudoinverse least-squares approximation

$$\mathbf{x} = \mathbf{A}_{\text{left}}^+\mathbf{b}$$



Pseudoinverse

- For matrix A with dimensions $N \times M$ with full rank
- Find solution that minimizes squared error: $\|Ax - b\|_2$
- Left pseudoinverse, for when $N > M$, (i.e., “tall”)

$$A_{\text{left}}^{-1} = (A^T A)^{-1} A^T \quad \text{s.t.} \quad A_{\text{left}}^{-1} A = I_n$$

- Right pseudoinverse, for when $N < M$, (i.e., “wide”)

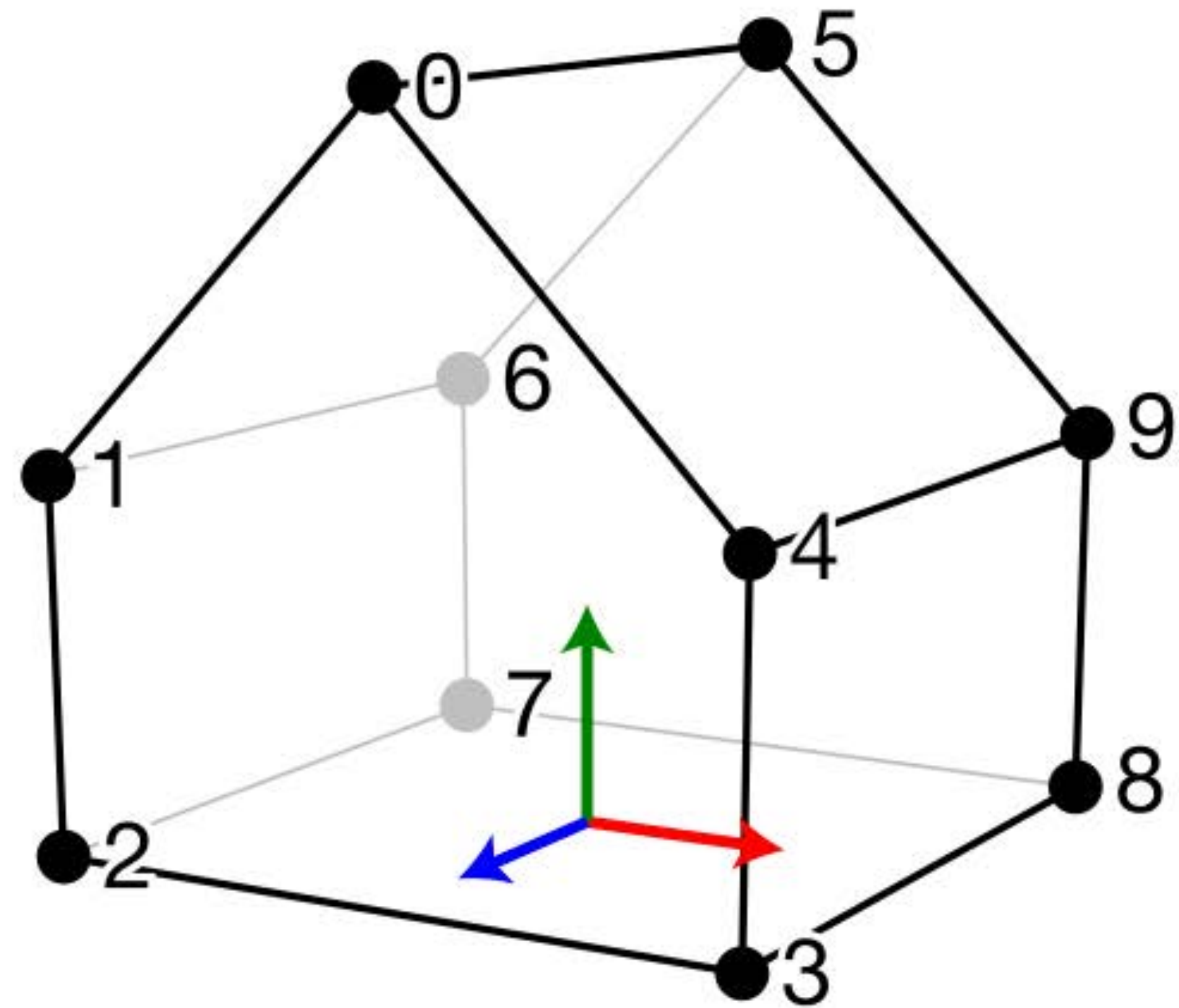
$$A_{\text{right}}^{-1} = A^T (A A^T)^{-1} \quad \text{s.t.} \quad A A_{\text{right}}^{-1} = I_m$$



How to define a Link Geometry



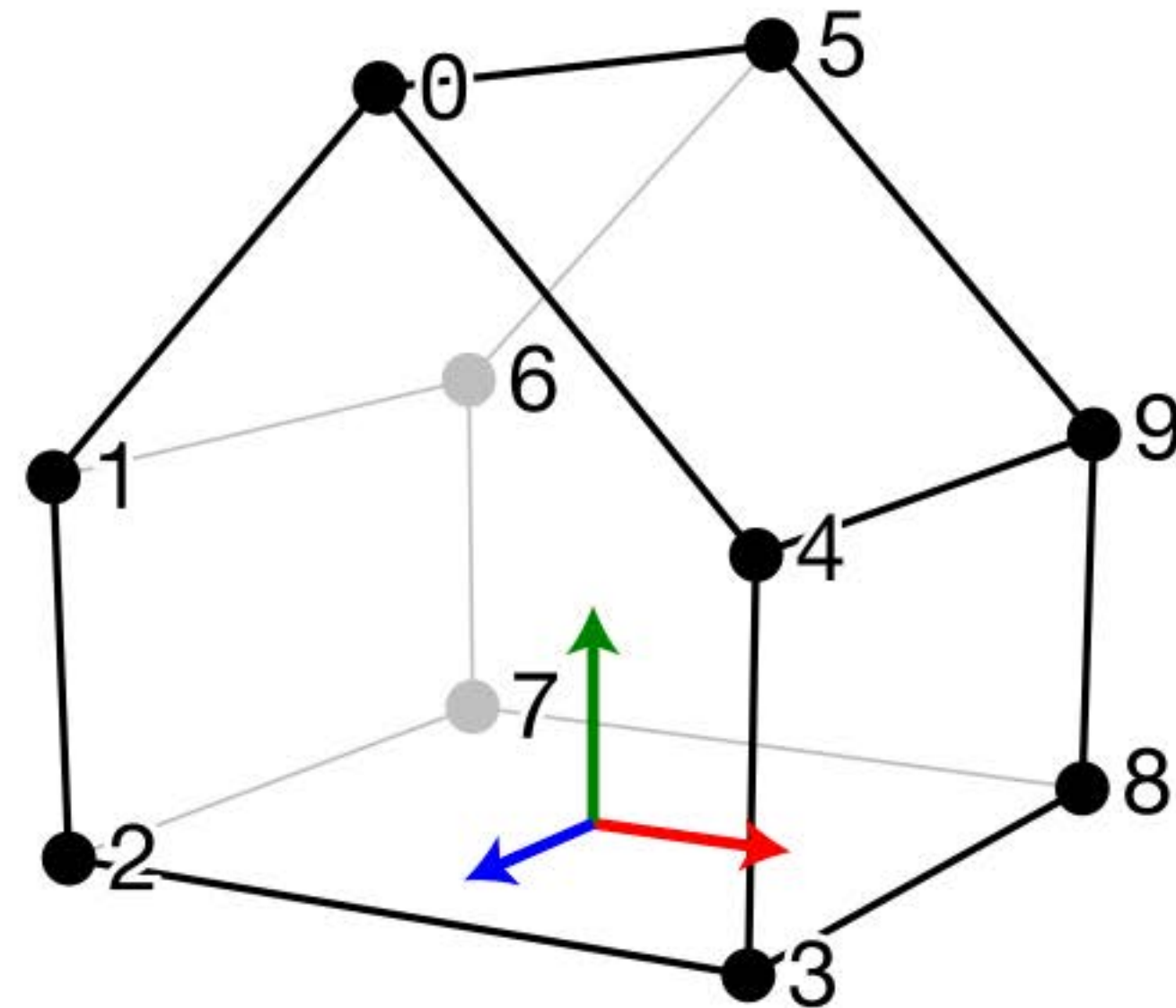
Link Geometry



<http://csc.lsu.edu/~kooima/courses/csc4356/>



Link Geometry



vertex index vertex location

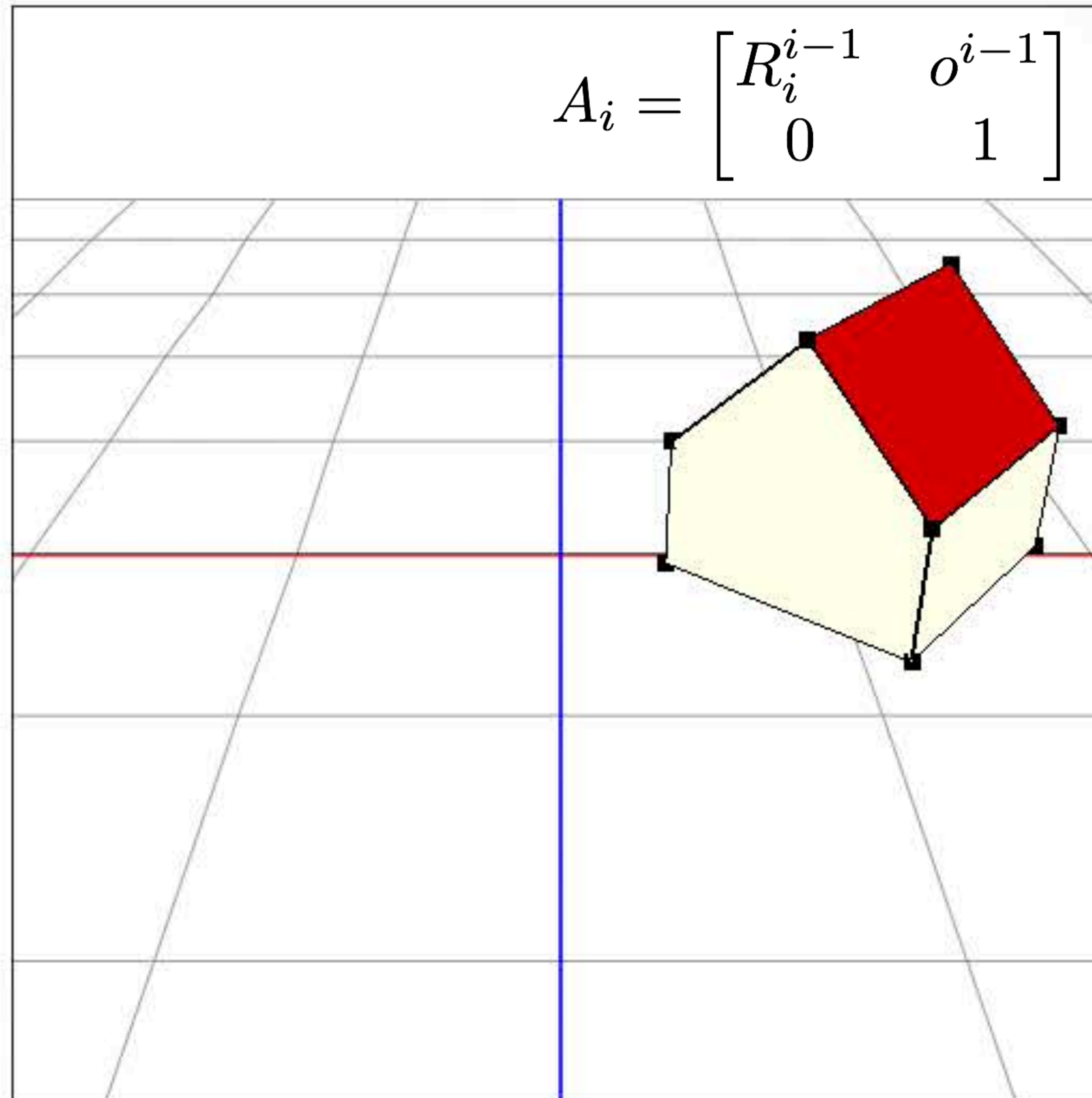
i	x	y	z
0	0.0	1.0	0.5
1	-0.5	0.5	0.5
2	-0.5	0.0	0.5
3	0.5	0.0	0.5
4	0.5	0.5	0.5
5	0.0	1.0	-0.5
6	-0.5	0.5	-0.5
7	-0.5	0.0	-0.5
8	0.5	0.0	-0.5
9	0.5	0.5	-0.5

Each robot link has a geometry specified as 3D vertices. Vertices are connected into faces of the object's surface. Vertices are defined wrt. the frame of the robots' link.

<http://csc.lsu.edu/~kooima/courses/csc4356/>



As the link frame moves, the geometry moves with it.

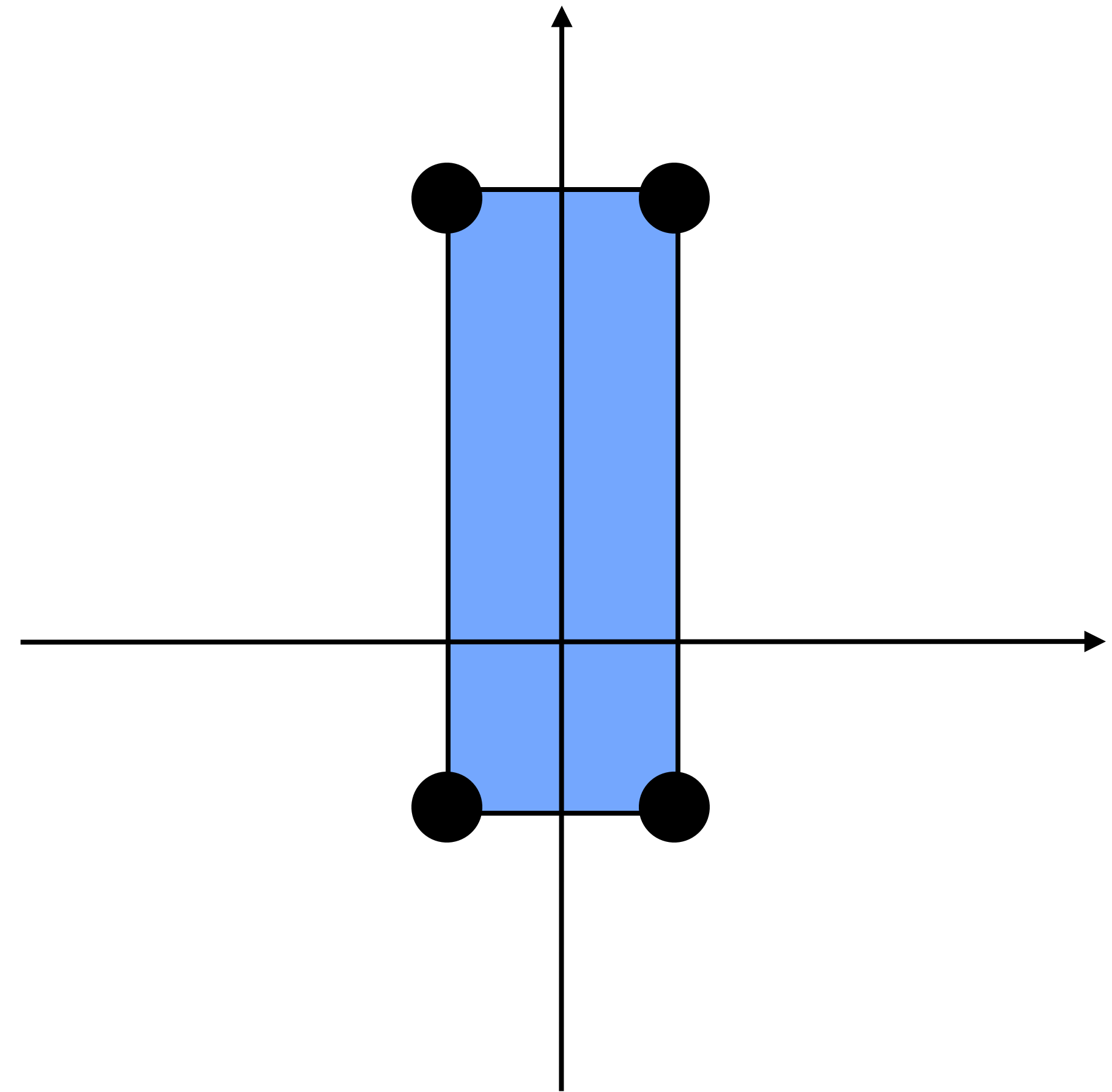


<http://csc.lsu.edu/~kooima/courses/csc4356/notes/04-transformation/transformation-composition-2.html>



2D Rotation

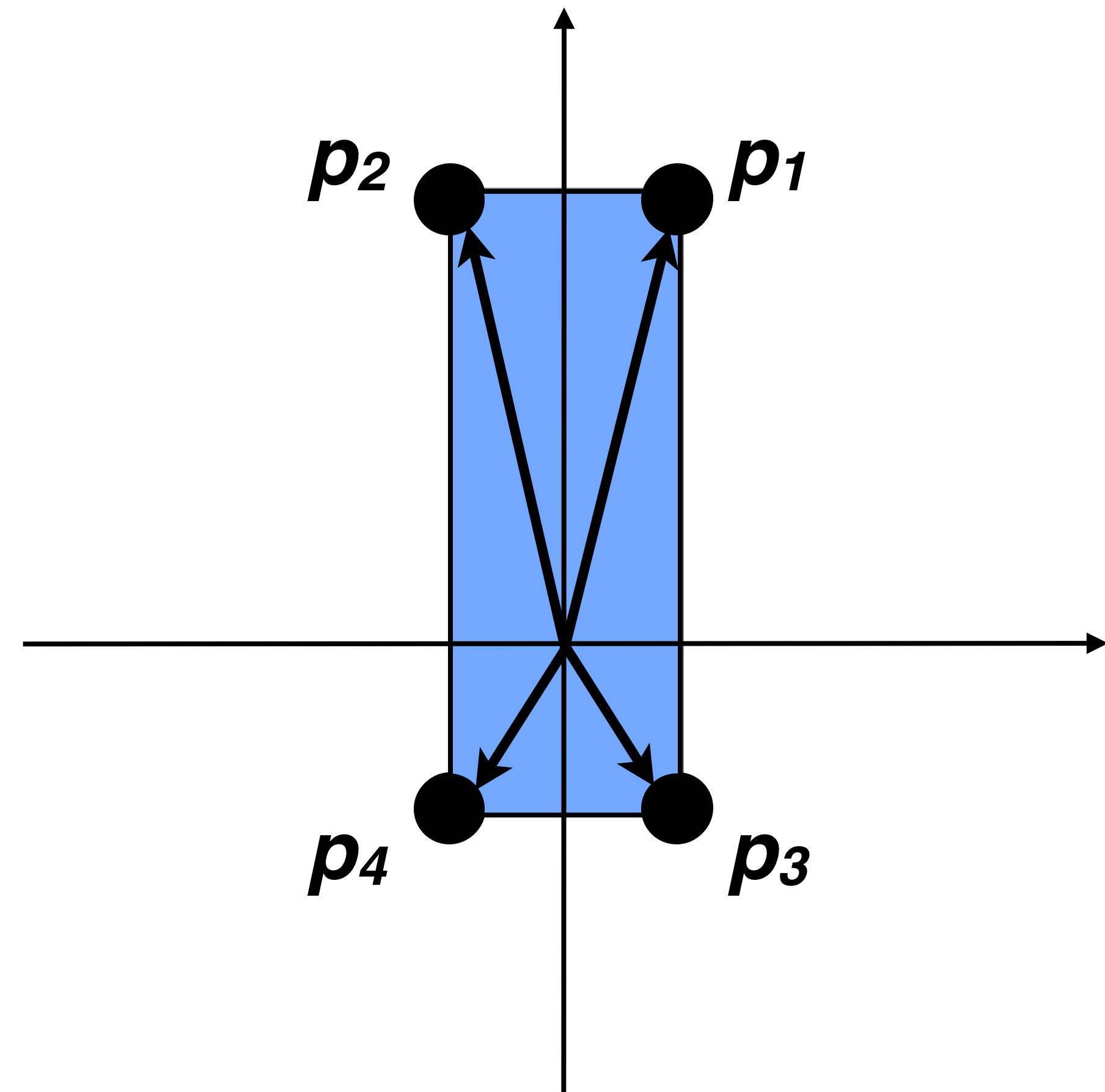
- Consider a link for a 2D robot with a box geometry of 4 vertices



2D Rotation

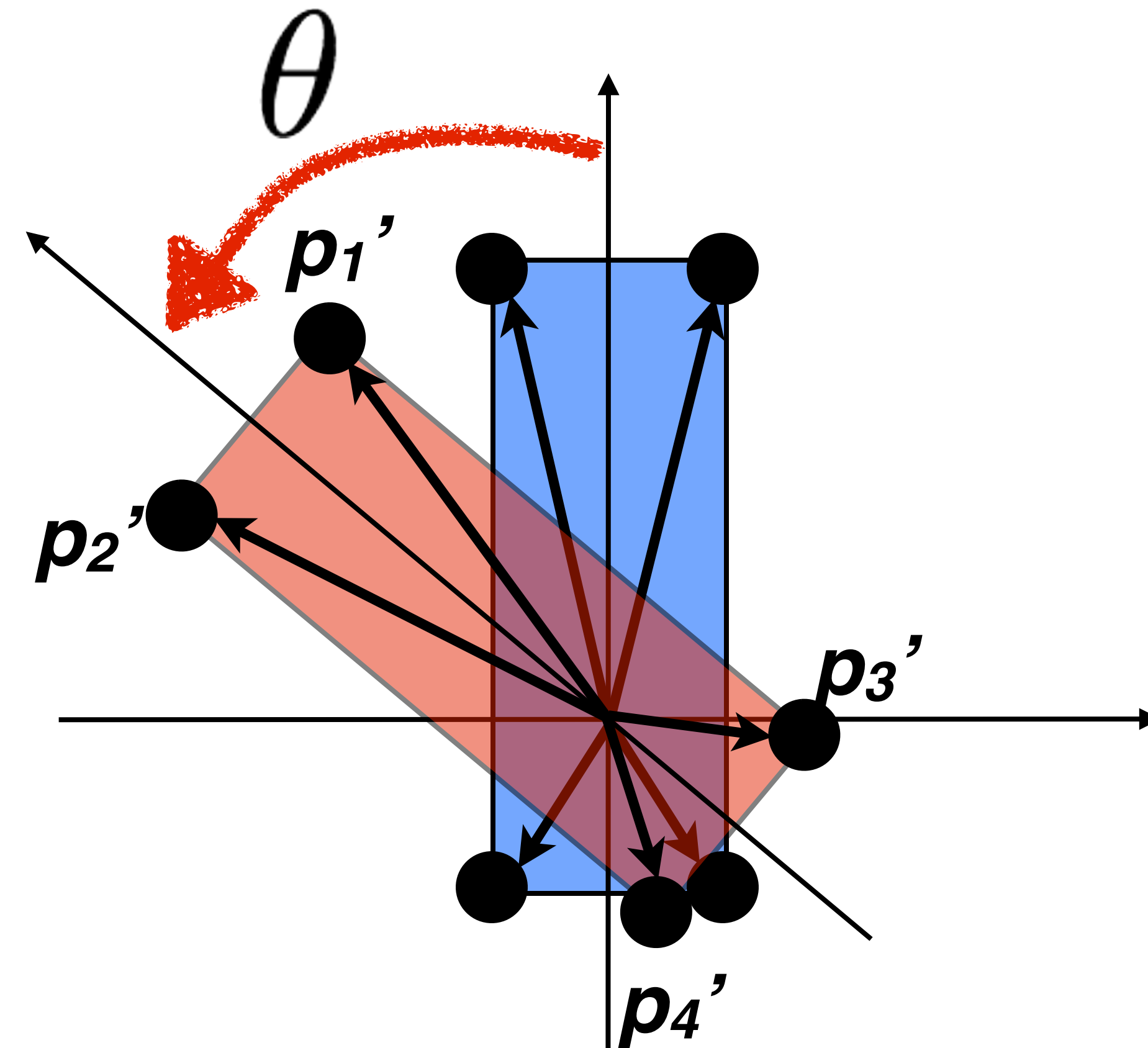
- Consider a link for a 2D robot with a box geometry of 4 vertices
- Vectors express position of vertices with respect to joint (at origin)

$$\mathbf{p}_i = [x_i, y_i]$$



2D Rotation

- Consider a link for a 2D robot with a box geometry of 4 vertices
- Vectors express position of vertices with respect to joint (at origin)
- How to rotate link geometry based on movement of the joint?



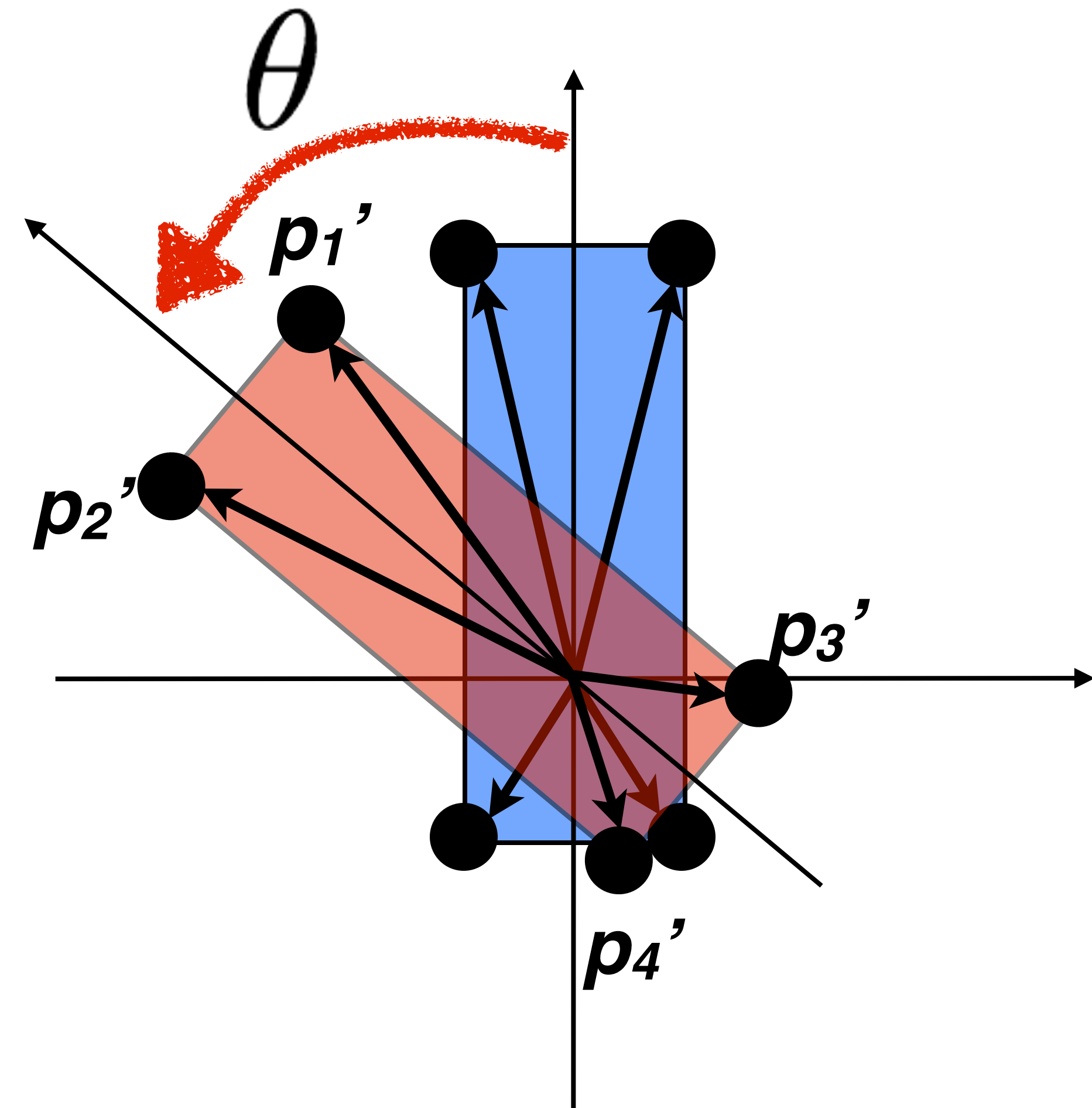
rotate about out-of-plane axis

2D Rotation

- Consider a link for a 2D robot with a box geometry of 4 vertices
- Vectors express position of vertices with respect to joint (at origin)
- How to rotate link geometry based on movement of the joint?

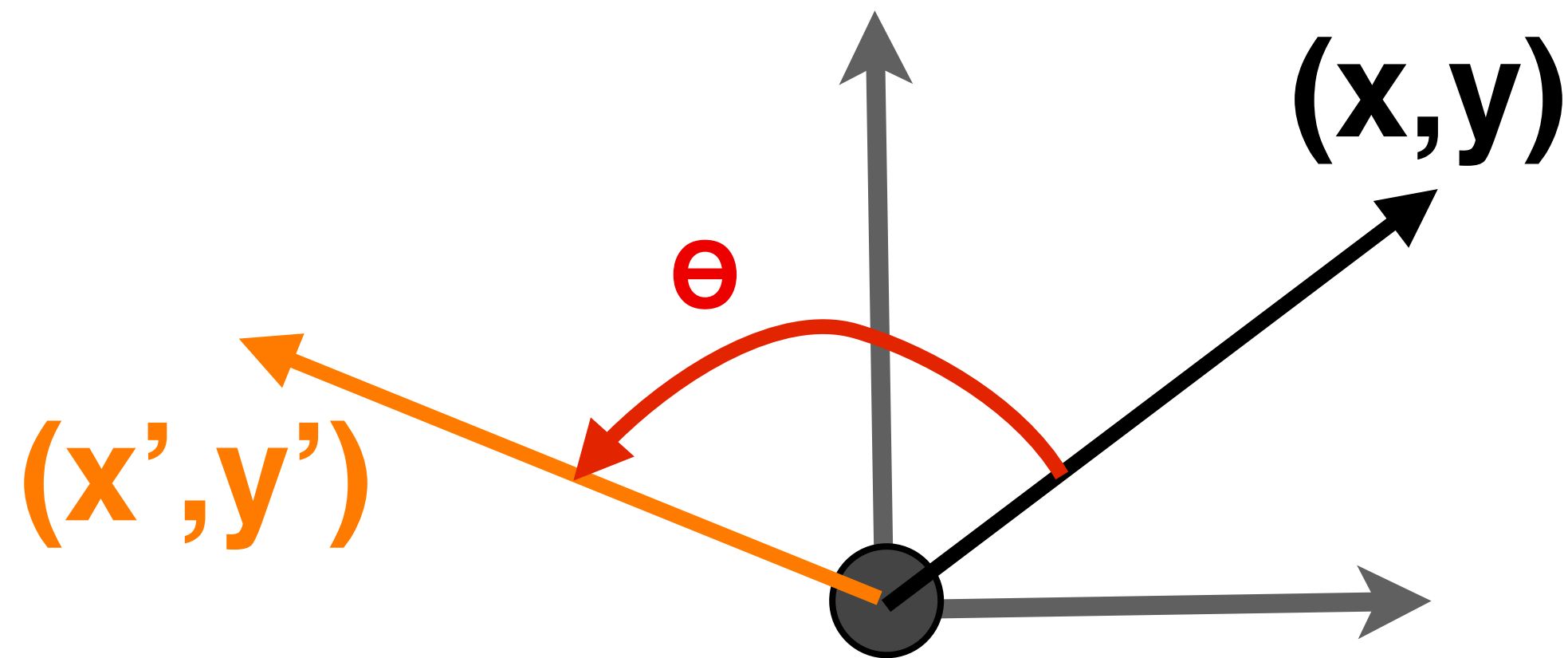
$$x' = x \cdot \cos(\theta) - y \cdot \sin(\theta)$$

$$y' = x \cdot \sin(\theta) + y \cdot \cos(\theta)$$



2D Rotation Matrix

(counterclockwise)

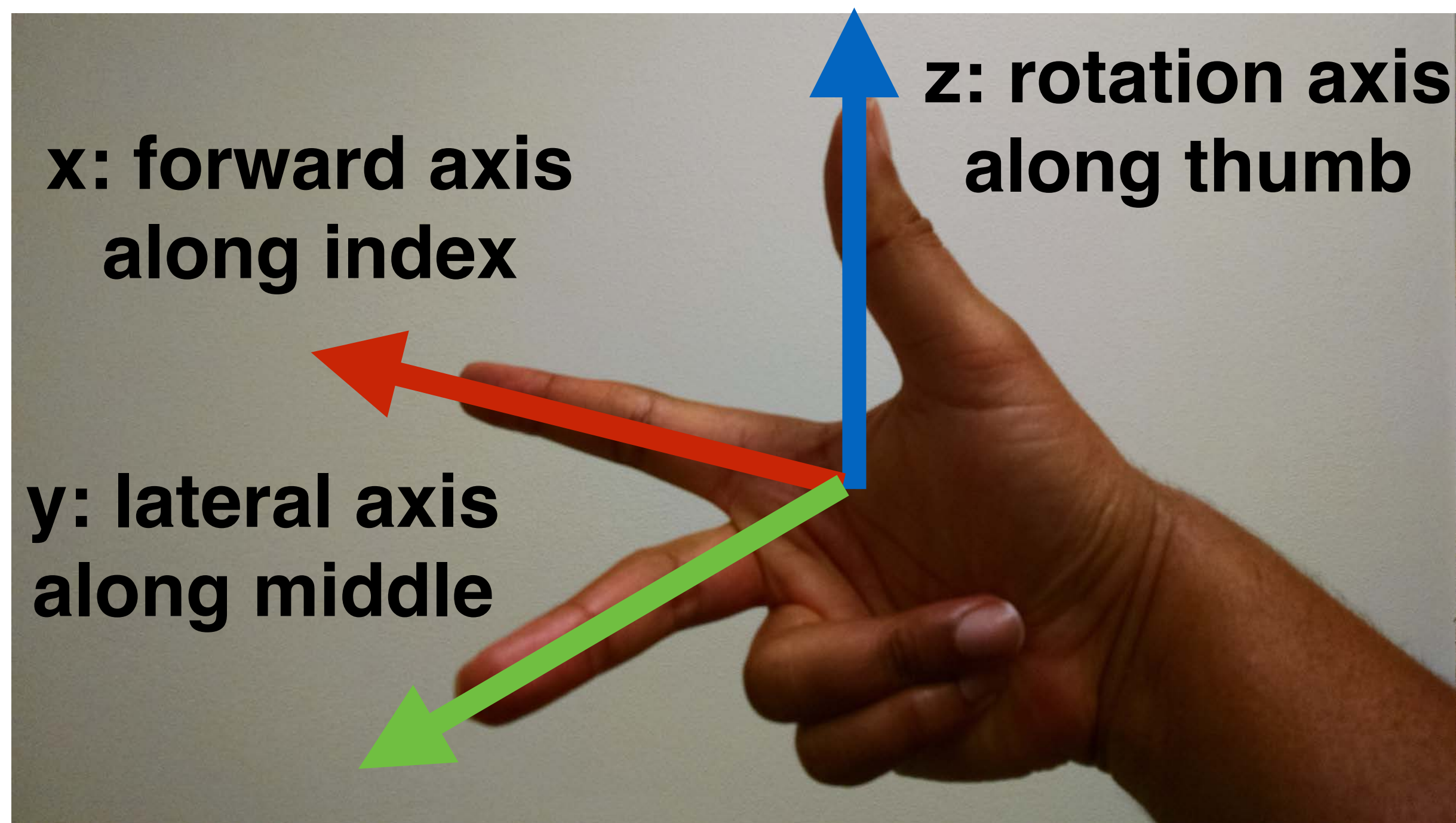


$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

$R(\theta)$

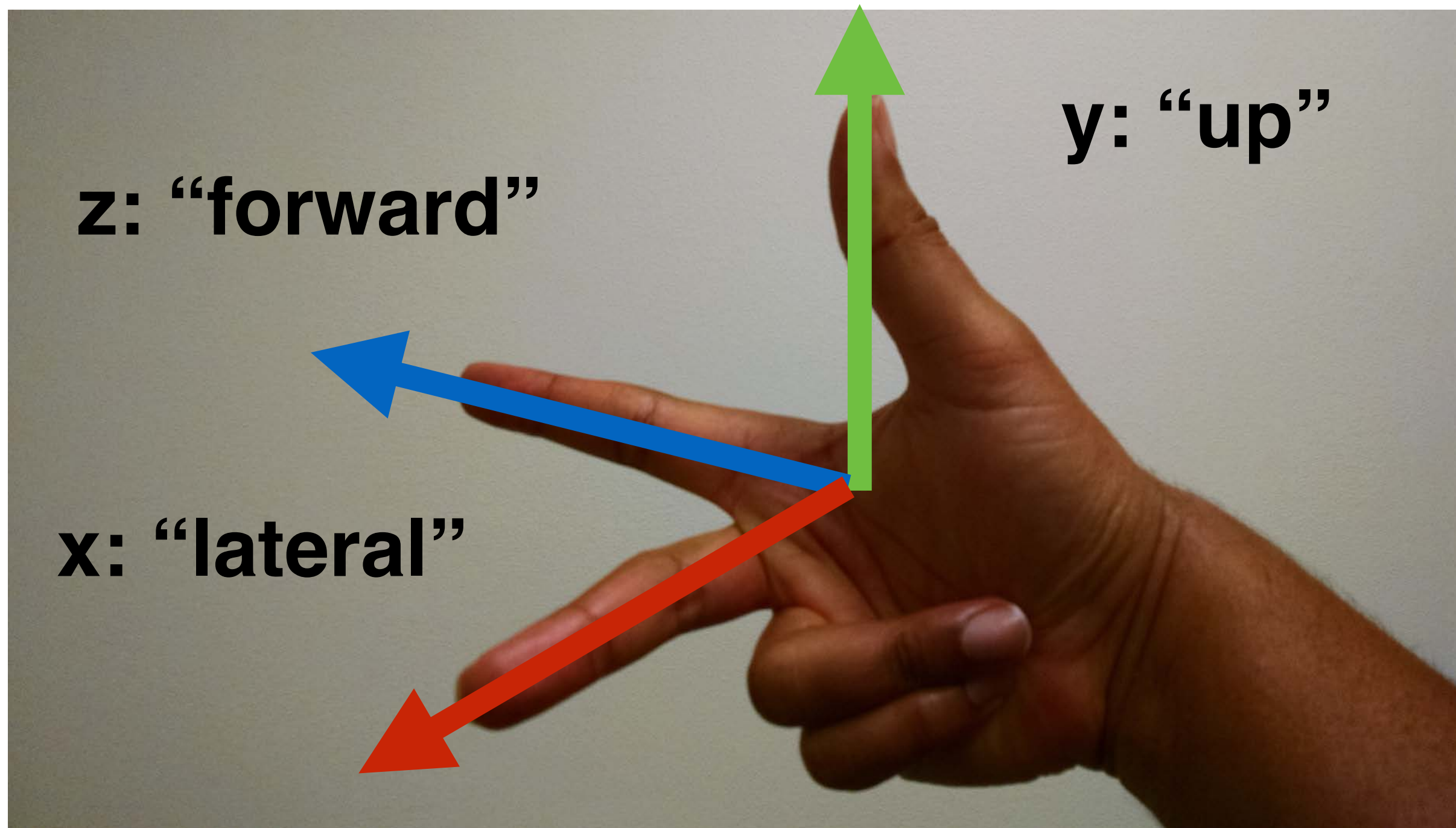
- Matrix multiply vector by 2D rotation matrix R
- Matrix parameterized by rotation angle θ
- Remember: this rotation is counterclockwise

Right-hand Rule

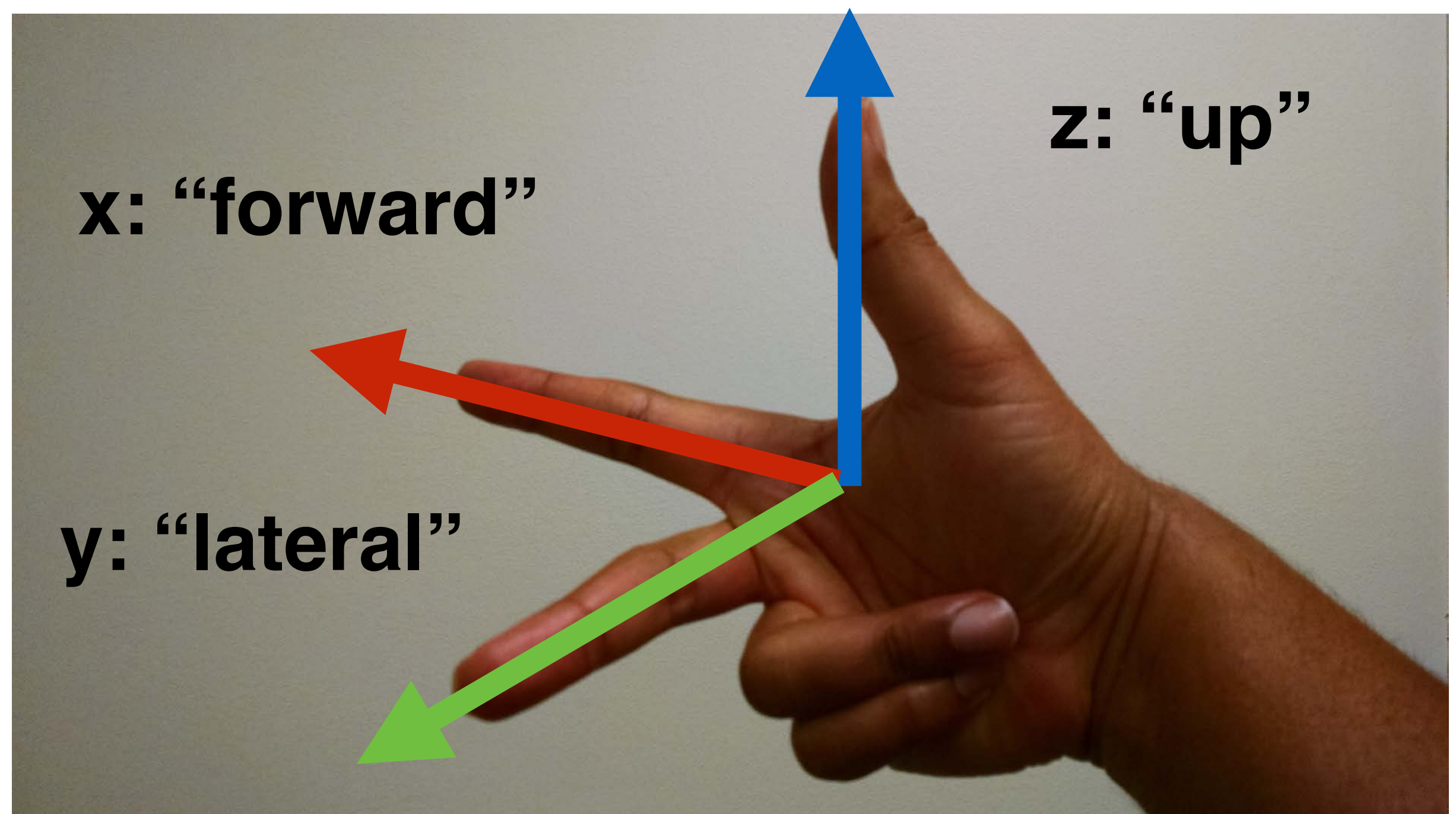


rotation occurs about axis from forward towards lateral,
or the “curl” of the fingers

Coordinate conventions



threejs and KinEval
(used in the browser)



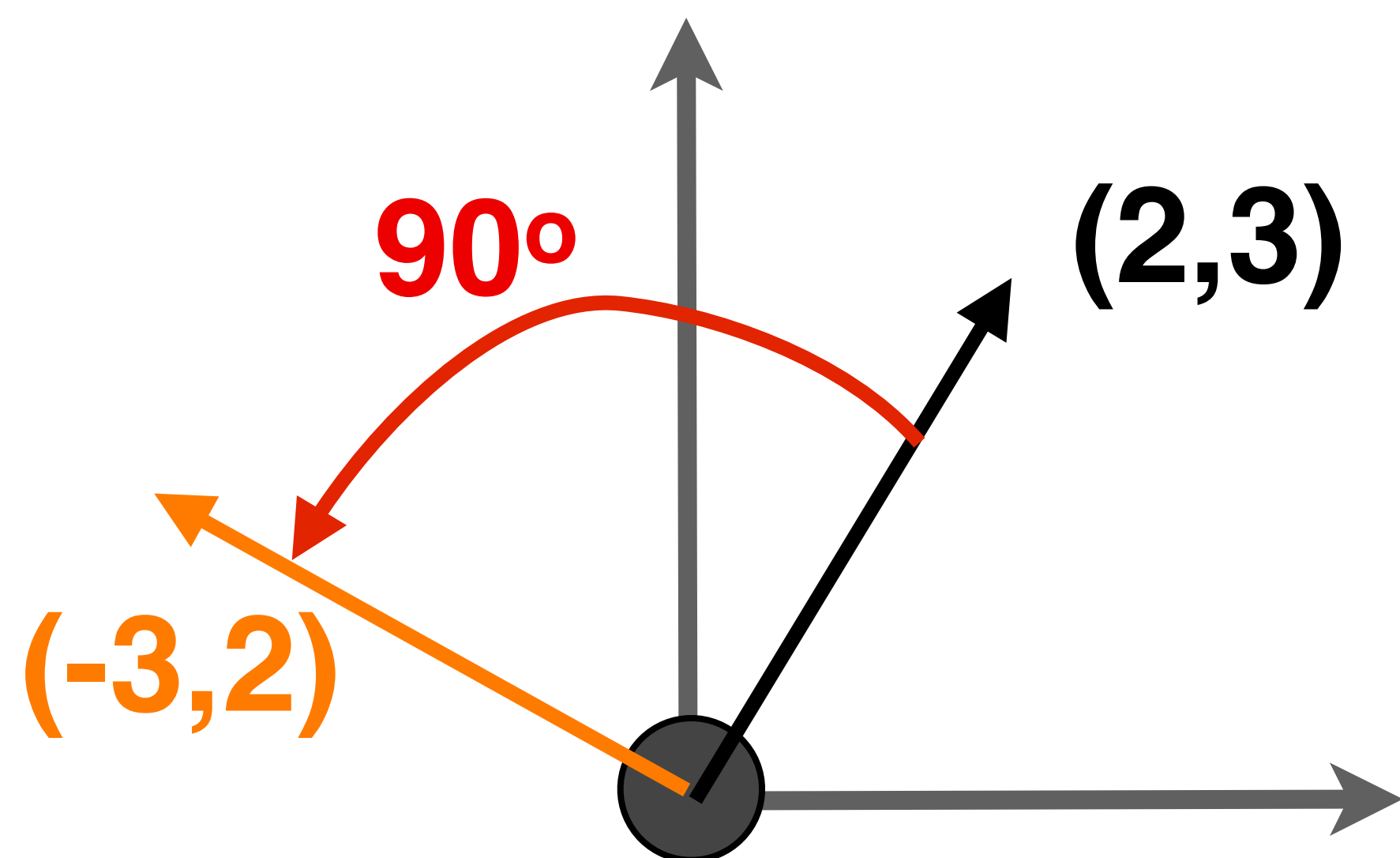
ROS and most of robotics
(used in URDF and rosbridge)

Checkpoint

- What is the 2D matrix for a rotation by 0 degrees?
- What is the 2D matrix for a rotation by 90 degrees?



Example



$$\begin{bmatrix} -3 \\ 2 \end{bmatrix}$$

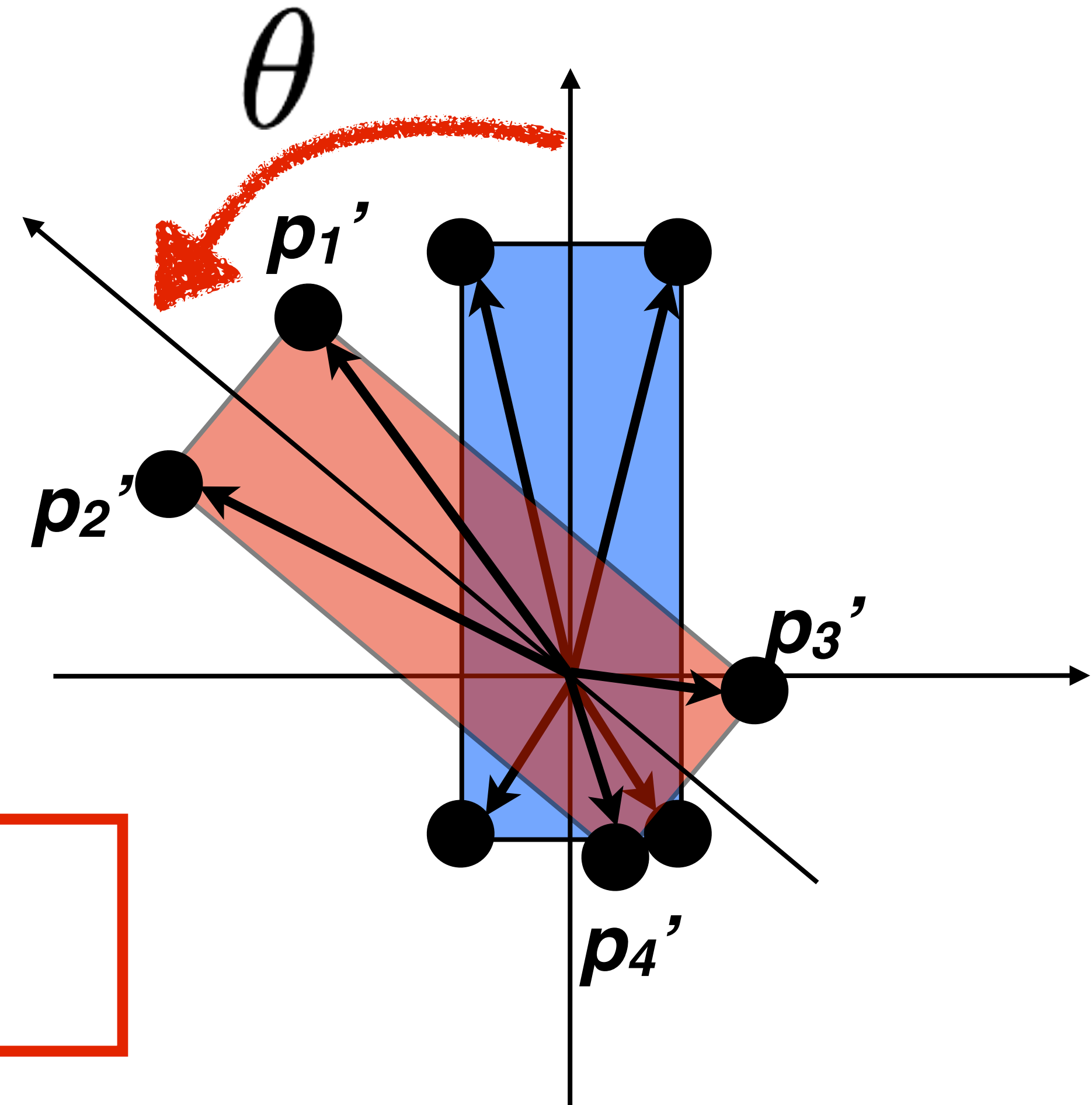
=

$$\begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 2 \\ 3 \end{bmatrix}$$

$R(90^\circ)$

$$\cos(90^\circ) = 0$$

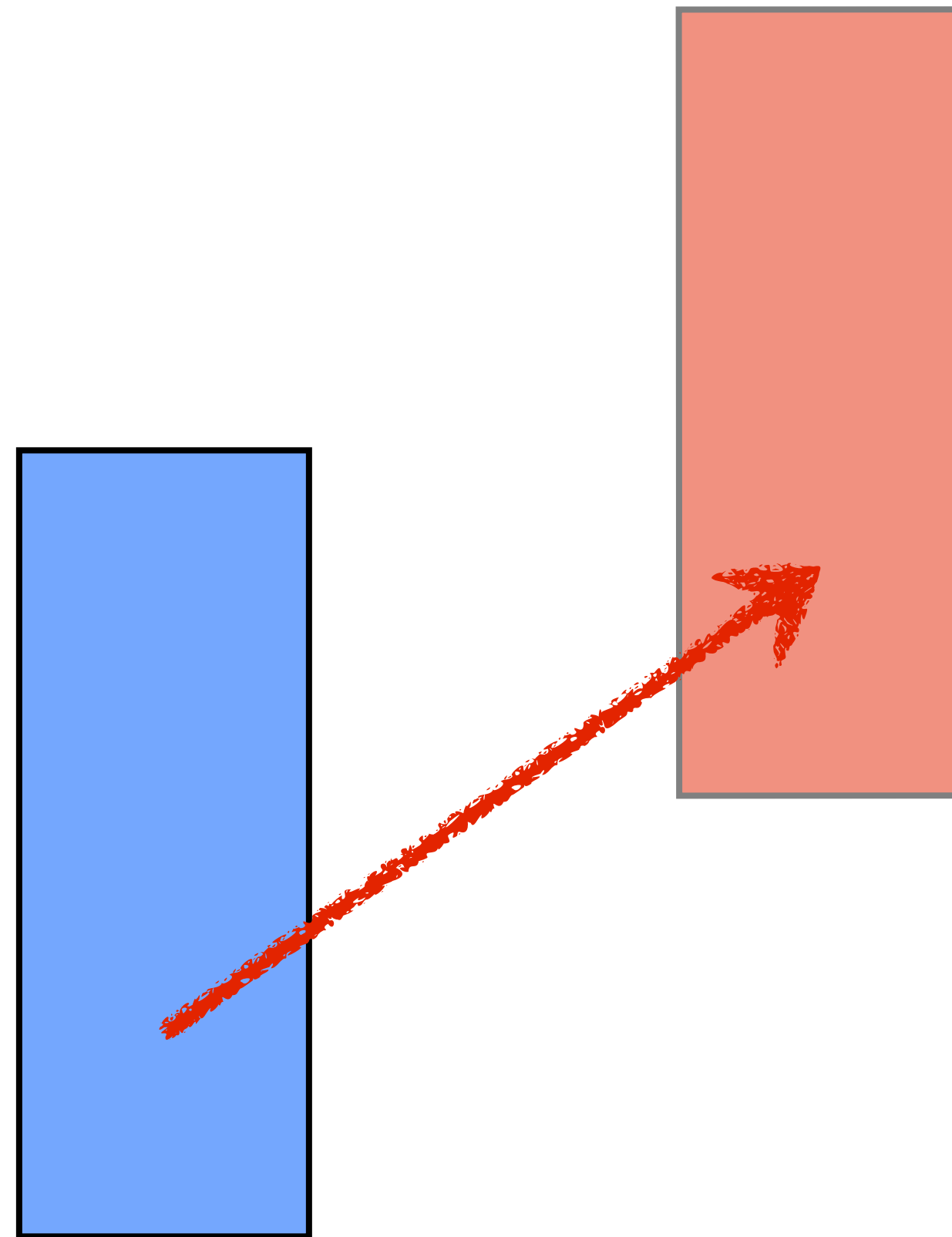
$$\sin(90^\circ) = 1$$



Note: one matrix multiply can transform all vertices

$$\begin{bmatrix} p'_{1x} & p'_{2x} & p'_{3x} & p'_{4x} \\ p'_{1y} & p'_{2y} & p'_{3y} & p'_{4y} \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} p_{1x} & p_{2x} & p_{3x} & p_{4x} \\ p_{1y} & p_{2y} & p_{3y} & p_{4y} \end{bmatrix}$$

We can rotate.
Can we also translate?

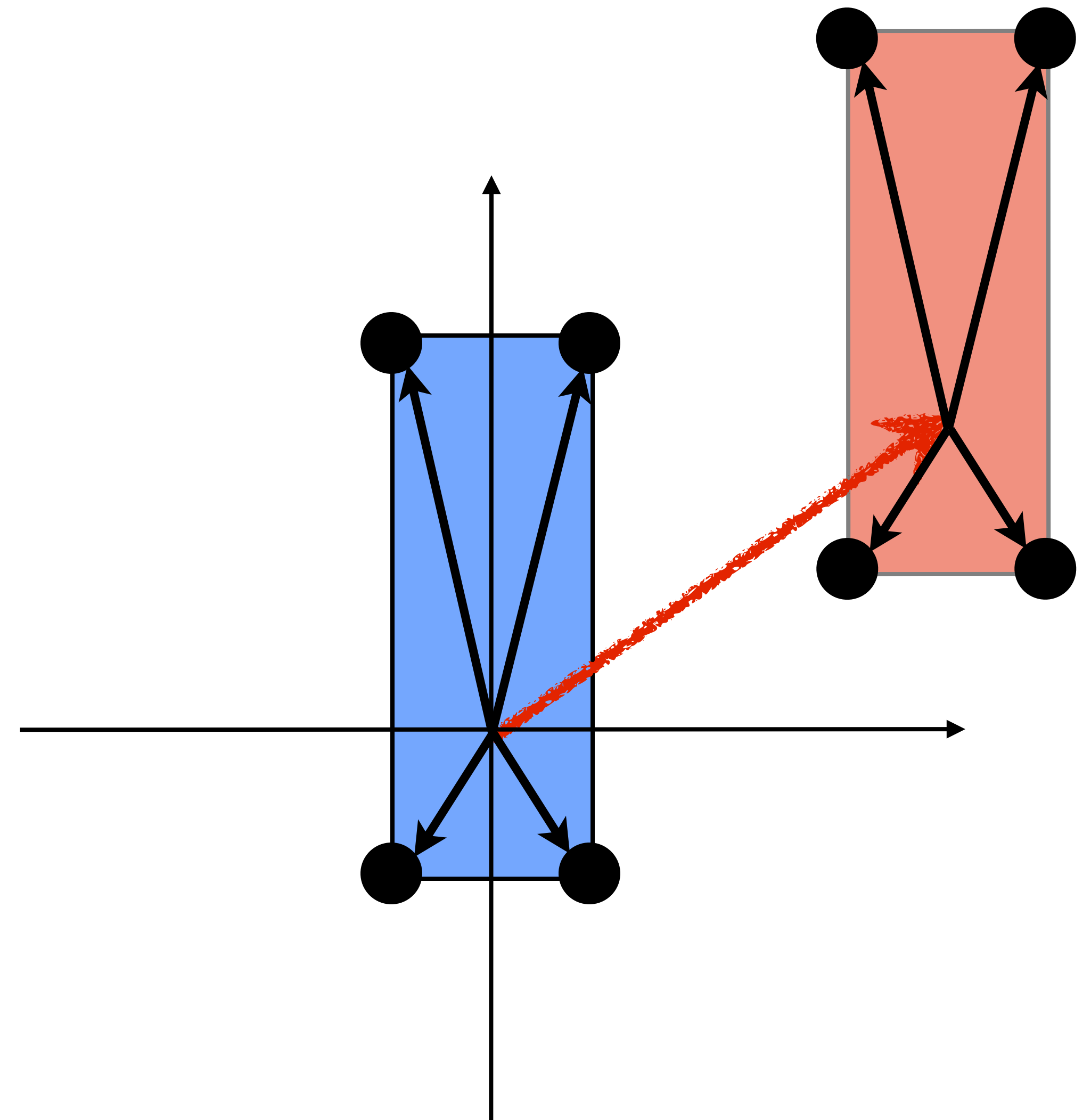


2D Translation

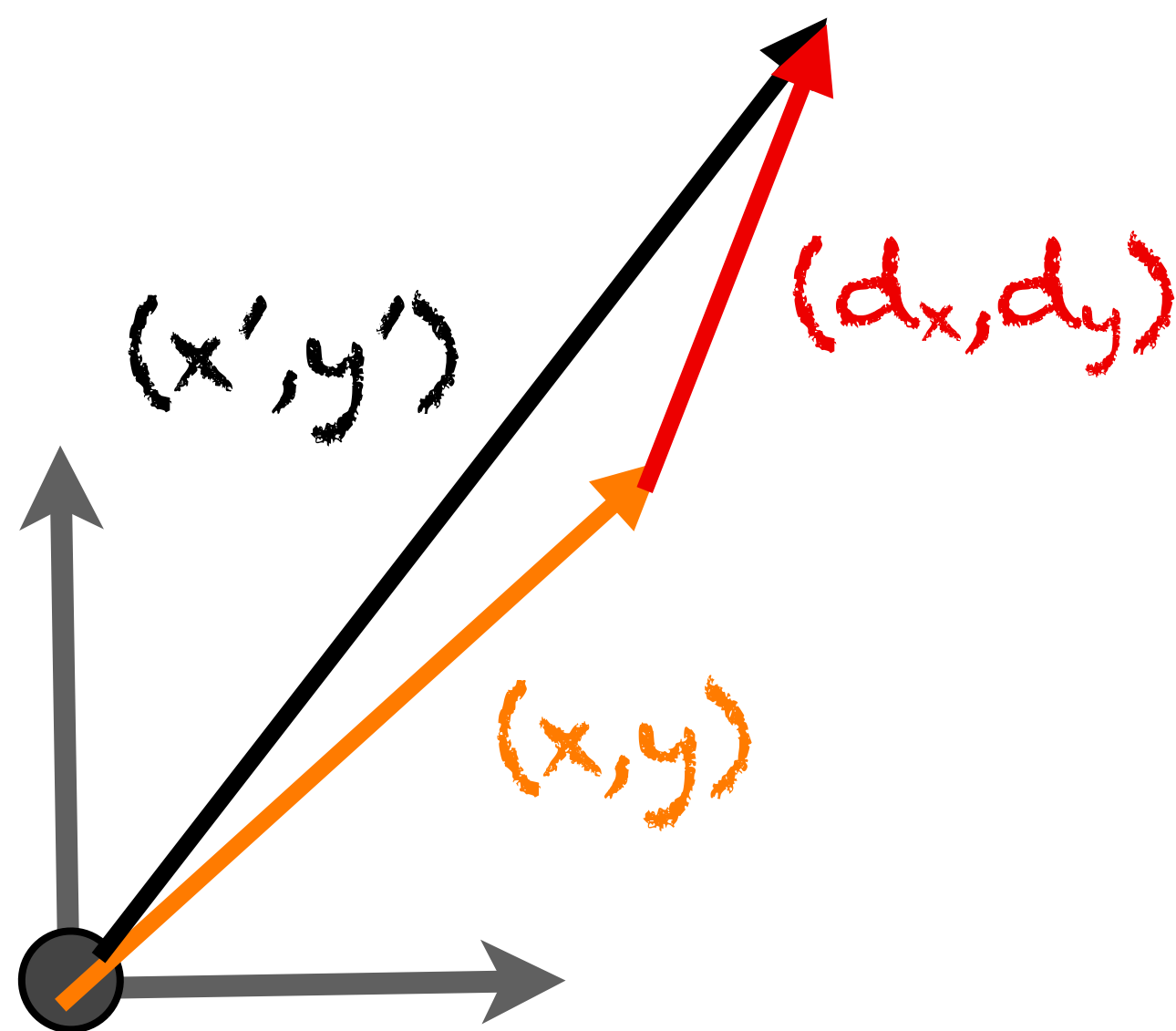
- Consider a link for a 2D robot with a box geometry of 4 vertices
- Vectors express position of vertices with respect to joint (at origin)
- How to translate link geometry to new location?

$$x' = x + d_x$$

$$y' = y + d_y$$



2D Translation Matrix



$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} x + d_x \\ y + d_y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & d_x \\ 0 & 1 & d_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

D(d_x, d_y)

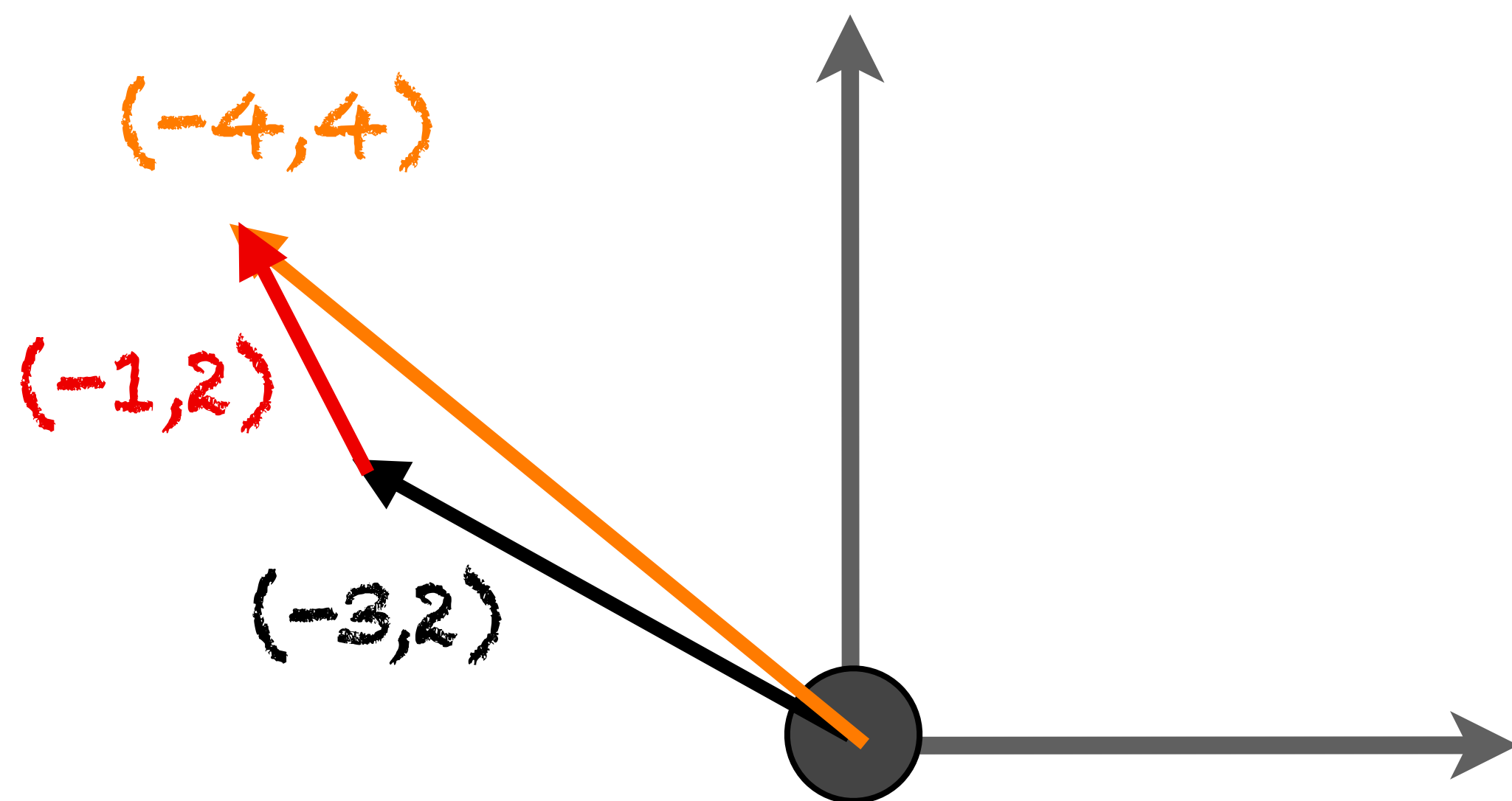
- Requires homogeneous coordinates
- 3D vector of 2D position concatenated with a 1
- A plane at $z=1$ in a three dimensional space
- Matrix parameterized by horizontal and vertical displacement (d_x, d_y)

Checkpoint

- What is the 2D matrix for a translation by $[-1, 2]$?



Example

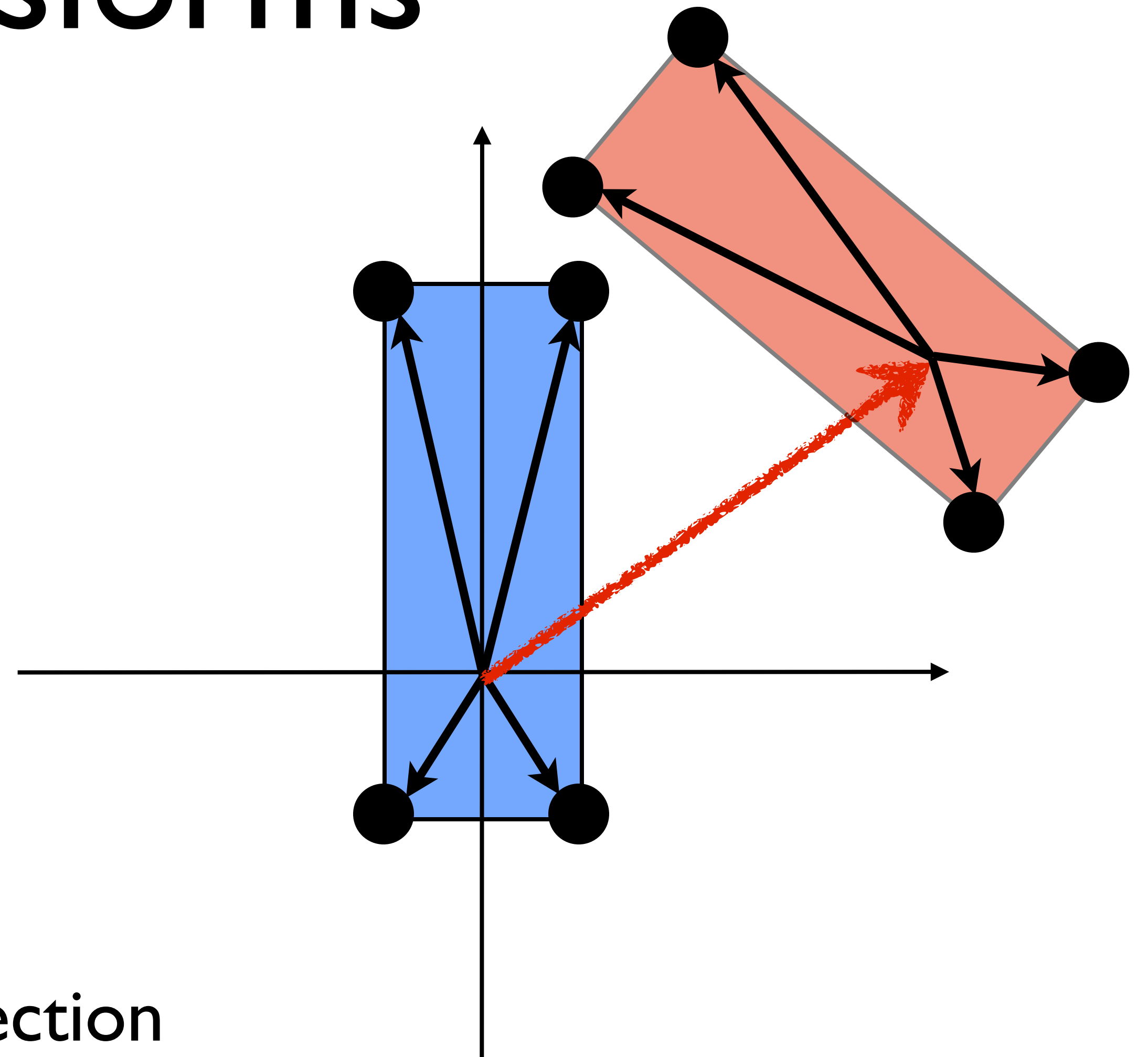


$$\begin{bmatrix} -4 \\ 4 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -3 \\ 2 \\ 1 \end{bmatrix}$$

$D(-1, 2)$

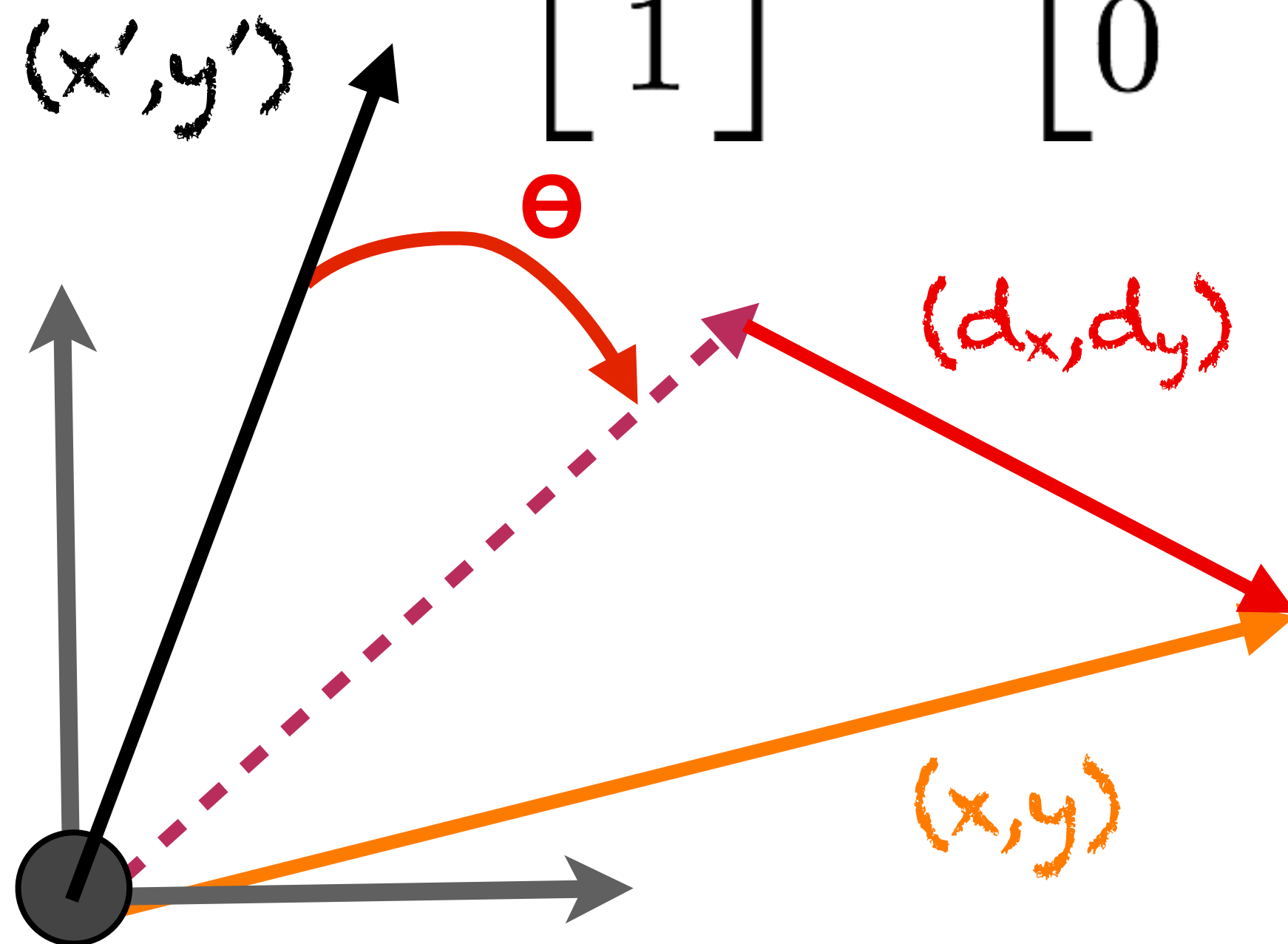
Rigid motions and Affine transforms

- Consider a link for a 2D robot with a box geometry of 4 vertices
- Vectors express position of vertices with respect to joint (at origin)
- How to both rotate and translate link geometry?
- Rigid motion: rotate then translate
- Affine transform: allows for rotation, translation, scaling, shearing, and reflection



Composition of Rotation and Translation

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & d_x \\ 0 & 1 & d_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

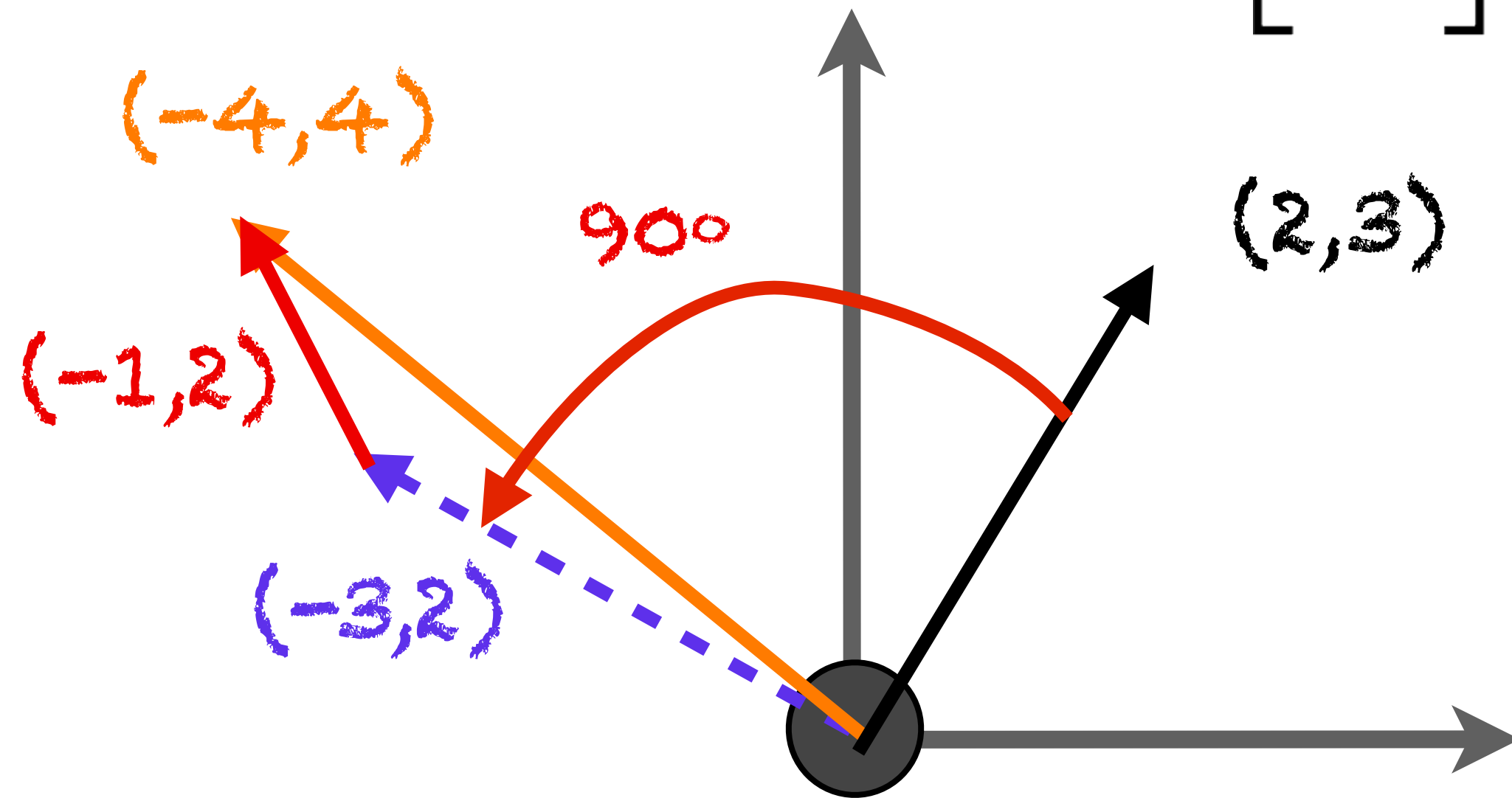


homogeneous rotation matrix

Example

$$\begin{bmatrix} -4 \\ 4 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 2 \\ 3 \\ 1 \end{bmatrix}$$

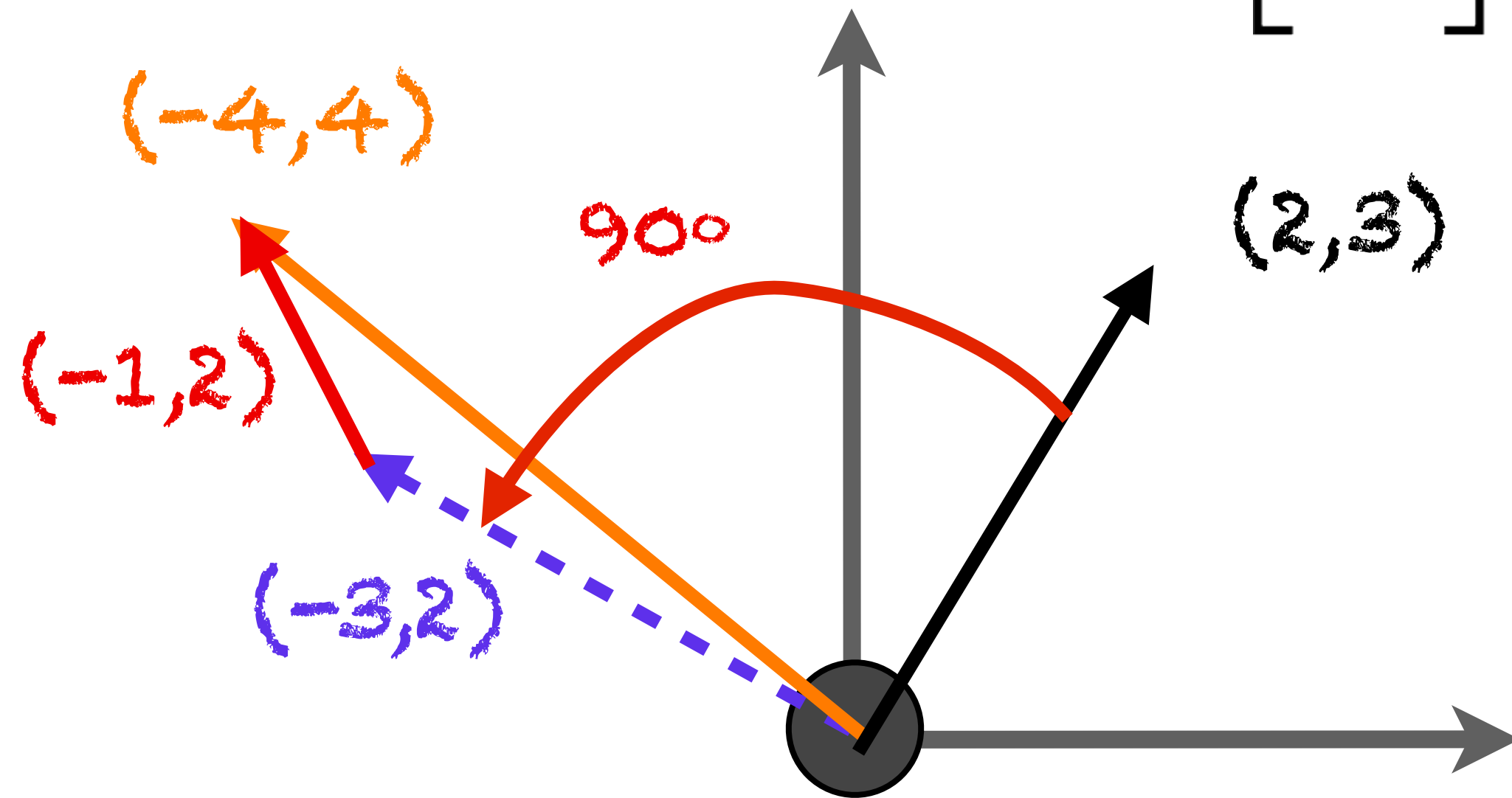
$D(-1,2)$ $R(90^\circ)$



Example

$$\begin{bmatrix} -4 \\ 4 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 2 \\ 3 \\ 1 \end{bmatrix}$$

$D(-1,2)$ $R(90^\circ)$

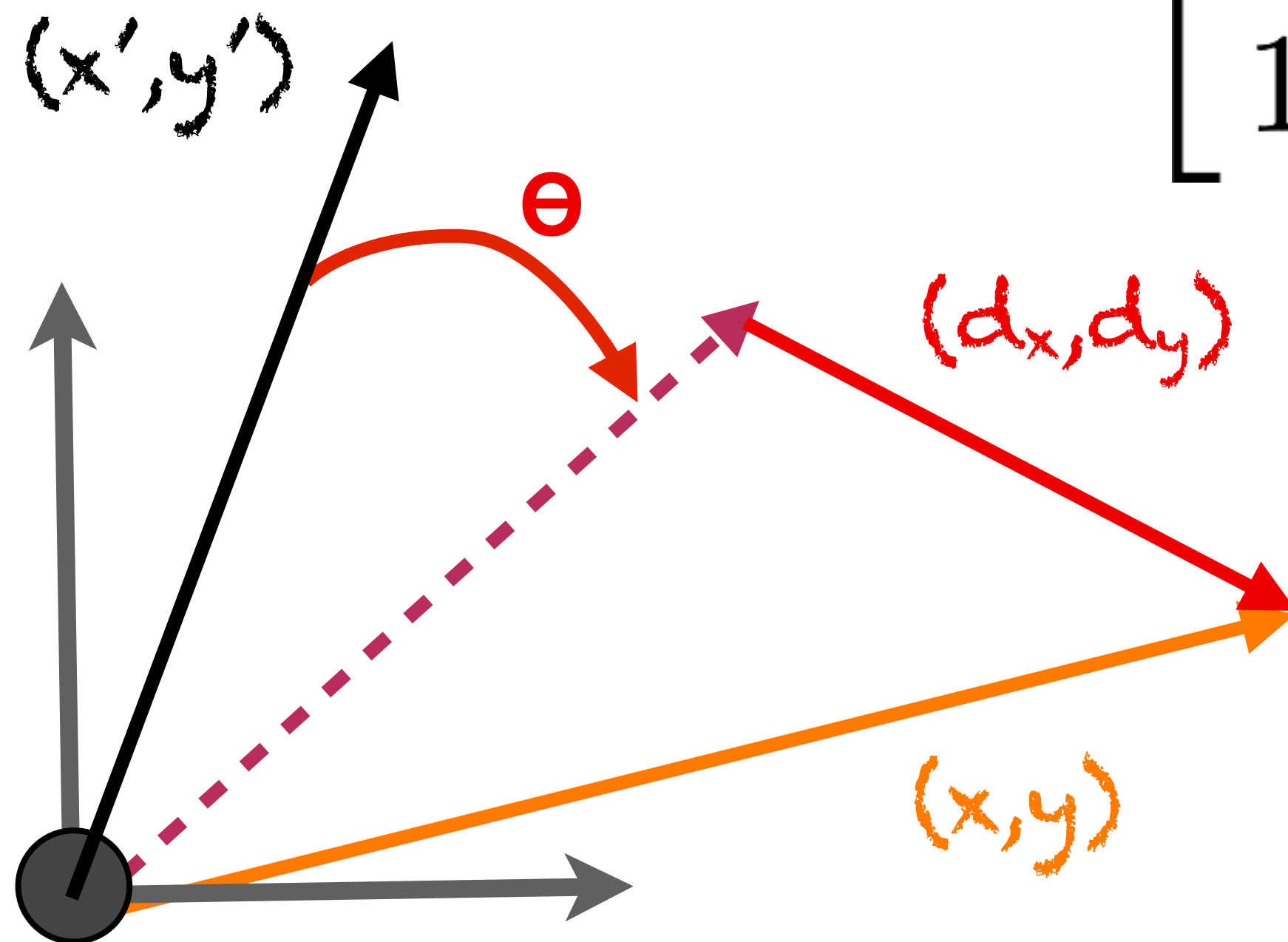


$$\begin{bmatrix} -4 \\ 4 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 & -1 & -1 \\ 1 & 0 & 2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 2 \\ 3 \\ 1 \end{bmatrix}$$

$D(-1,2)R(90^\circ)$

Homogeneous Transform: Composition of Rotation and Translation

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & d_x \\ \sin(\theta) & \cos(\theta) & d_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$



$$A_i = \begin{bmatrix} R_i^{i-1} & o^{i-1} \\ 0 & 1 \end{bmatrix}$$

Homogeneous Transform

defines SE(2): Special Euclidean Group 2

$$H = \begin{bmatrix} R_{00} & R_{01} & d_x \\ R_{10} & R_{11} & d_y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}_{2 \times 2} & \mathbf{d}_{2 \times 1} \\ \mathbf{0}_{1 \times 2} & 1 \end{bmatrix}$$



Homogeneous Transform

defines SE(2): Special Euclidean Group 2

$$H = \begin{bmatrix} R_{00} & R_{01} & d_x \\ R_{10} & R_{11} & d_y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}_{2 \times 2} & \mathbf{d}_{2 \times 1} \\ \mathbf{0}_{1 \times 2} & 1 \end{bmatrix}$$

H \in *SE*(2)



Homogeneous Transform

defines SE(2): Special Euclidean Group 2

$$H = \begin{bmatrix} R_{00} & R_{01} & d_x \\ R_{10} & R_{11} & d_y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}_{2 \times 2} & \mathbf{d}_{2 \times 1} \\ \mathbf{0}_{1 \times 2} & 1 \end{bmatrix}$$

$$H \in SE(2) \quad \mathbf{R}_{2 \times 2} \in SO(2)$$



Homogeneous Transform

defines SE(2): Special Euclidean Group 2

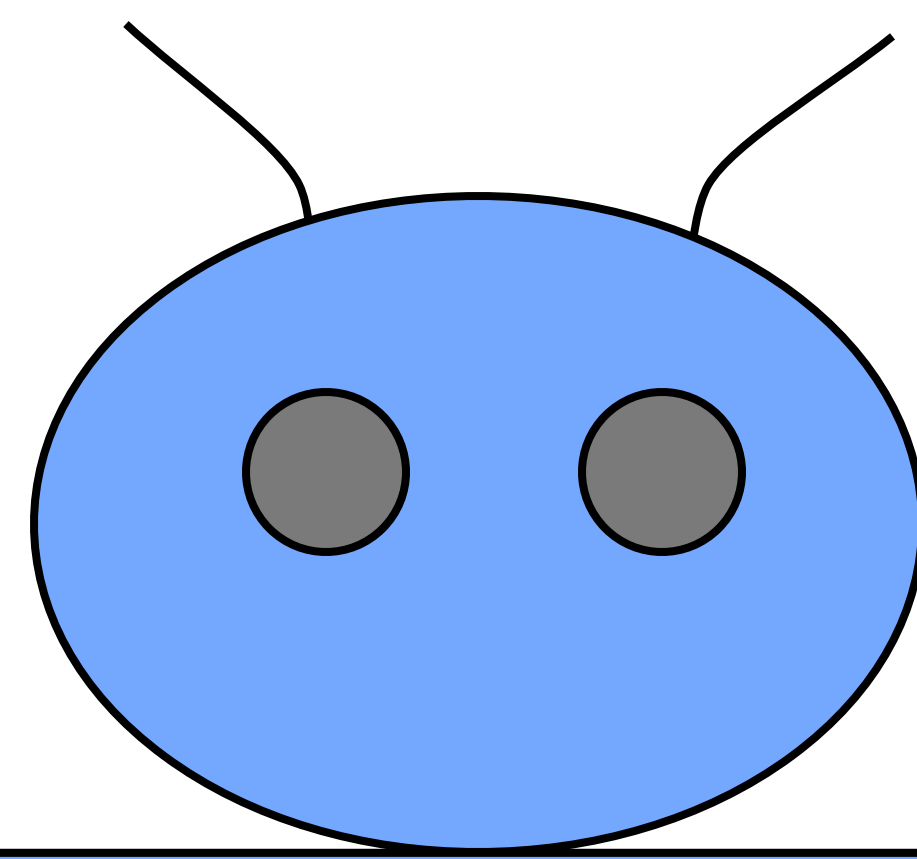
$$H = \begin{bmatrix} R_{00} & R_{01} & d_x \\ R_{10} & R_{11} & d_y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}_{2 \times 2} & \mathbf{d}_{2 \times 1} \\ \mathbf{0}_{1 \times 2} & 1 \end{bmatrix}$$

$H \in SE(2)$ $\mathbf{R}_{2 \times 2} \in SO(2)$ $\mathbf{d}_{2 \times 1} \in \mathbb{R}^2$

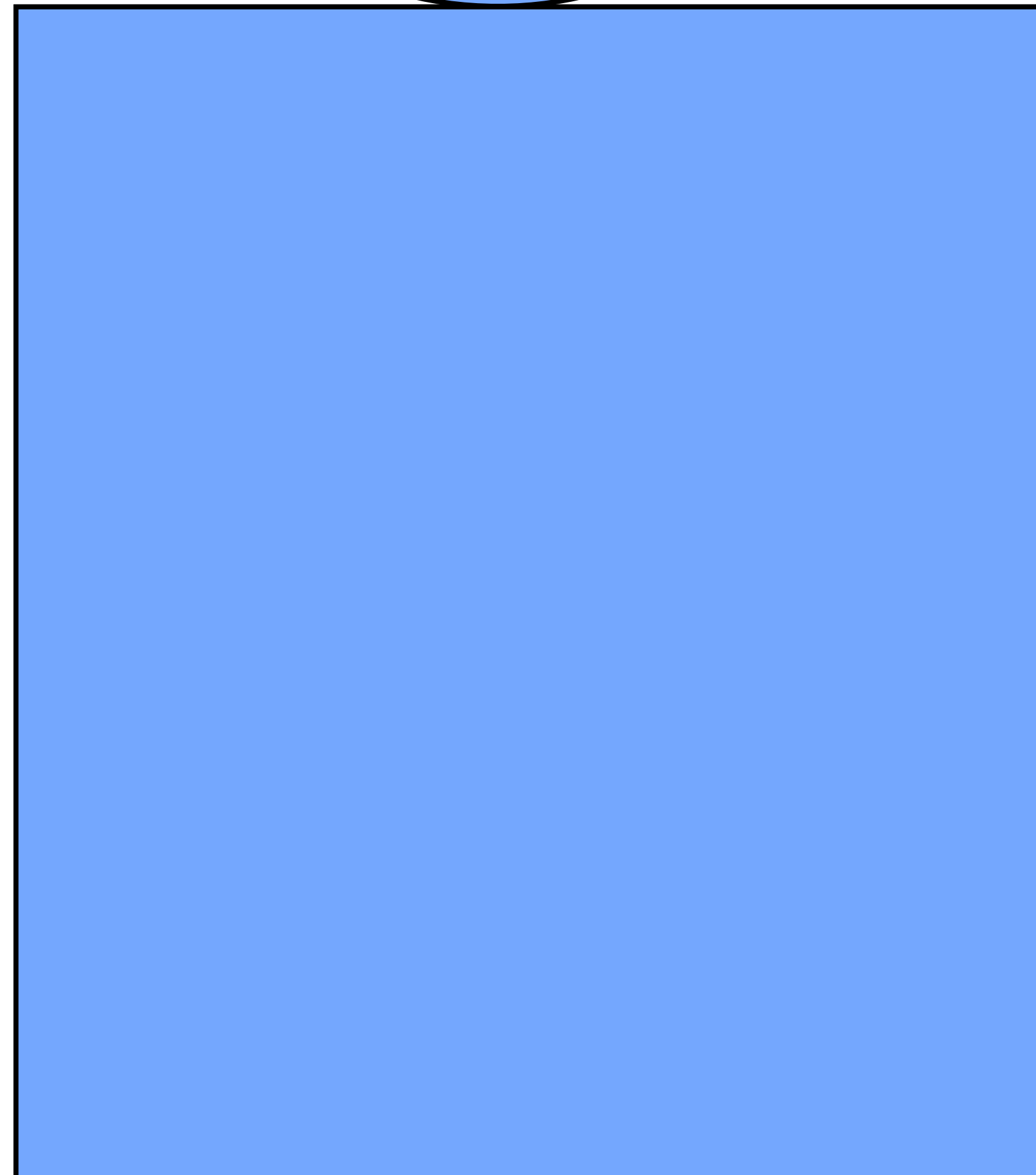


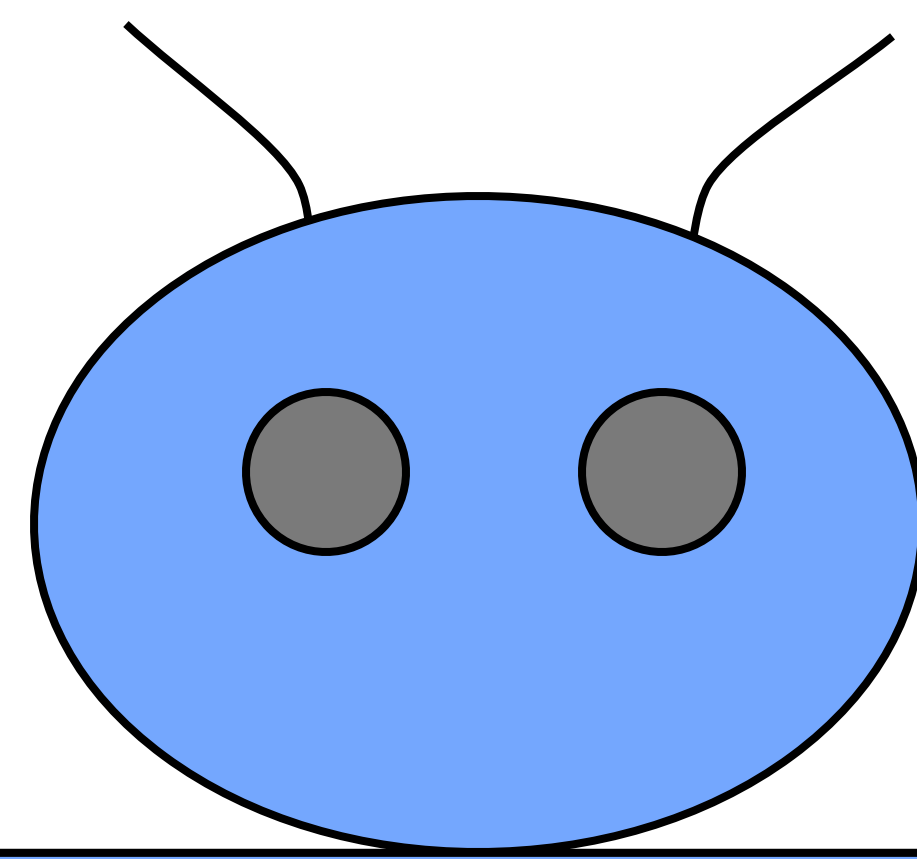
Example:
Let's put an arm link on Boxy



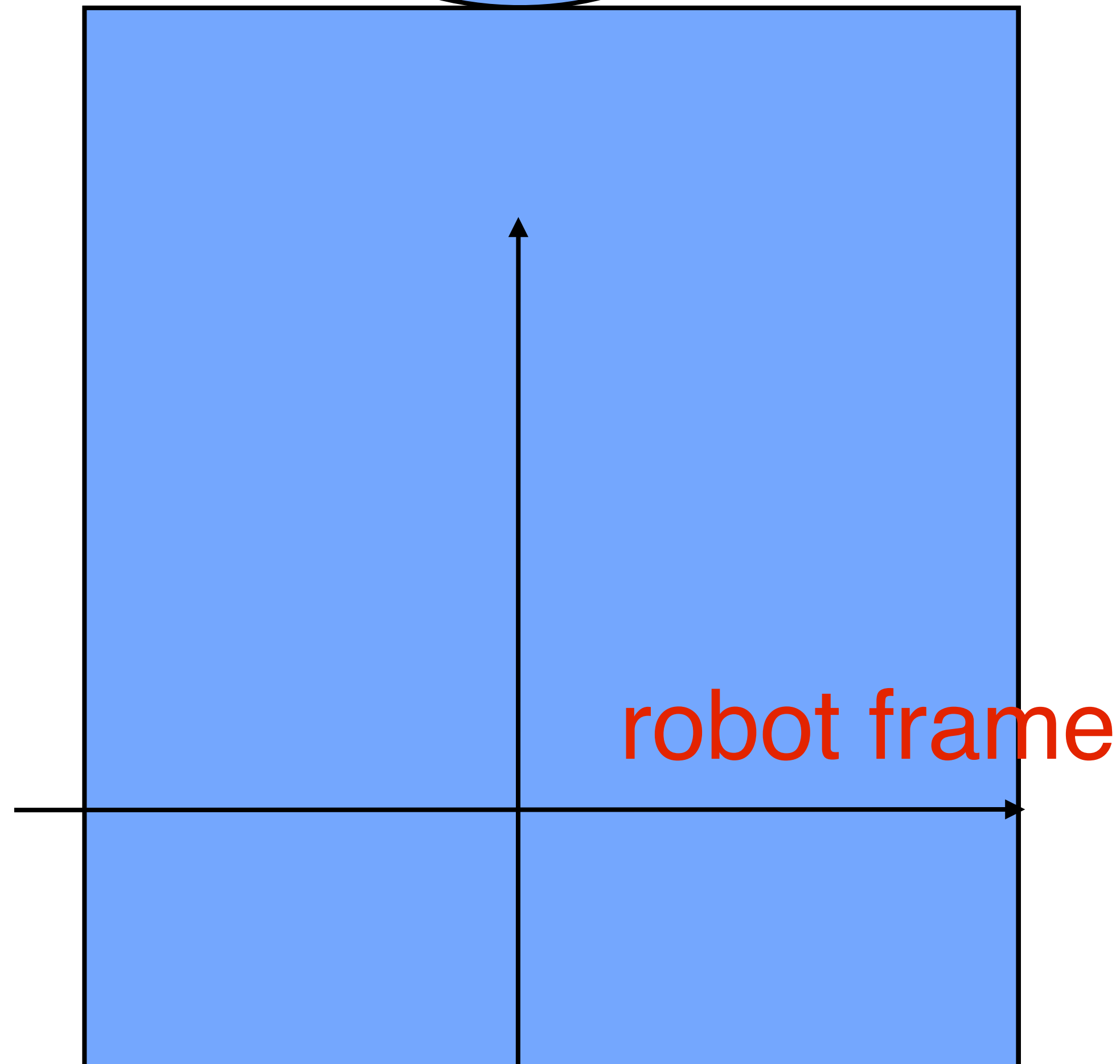


Boxy the robot

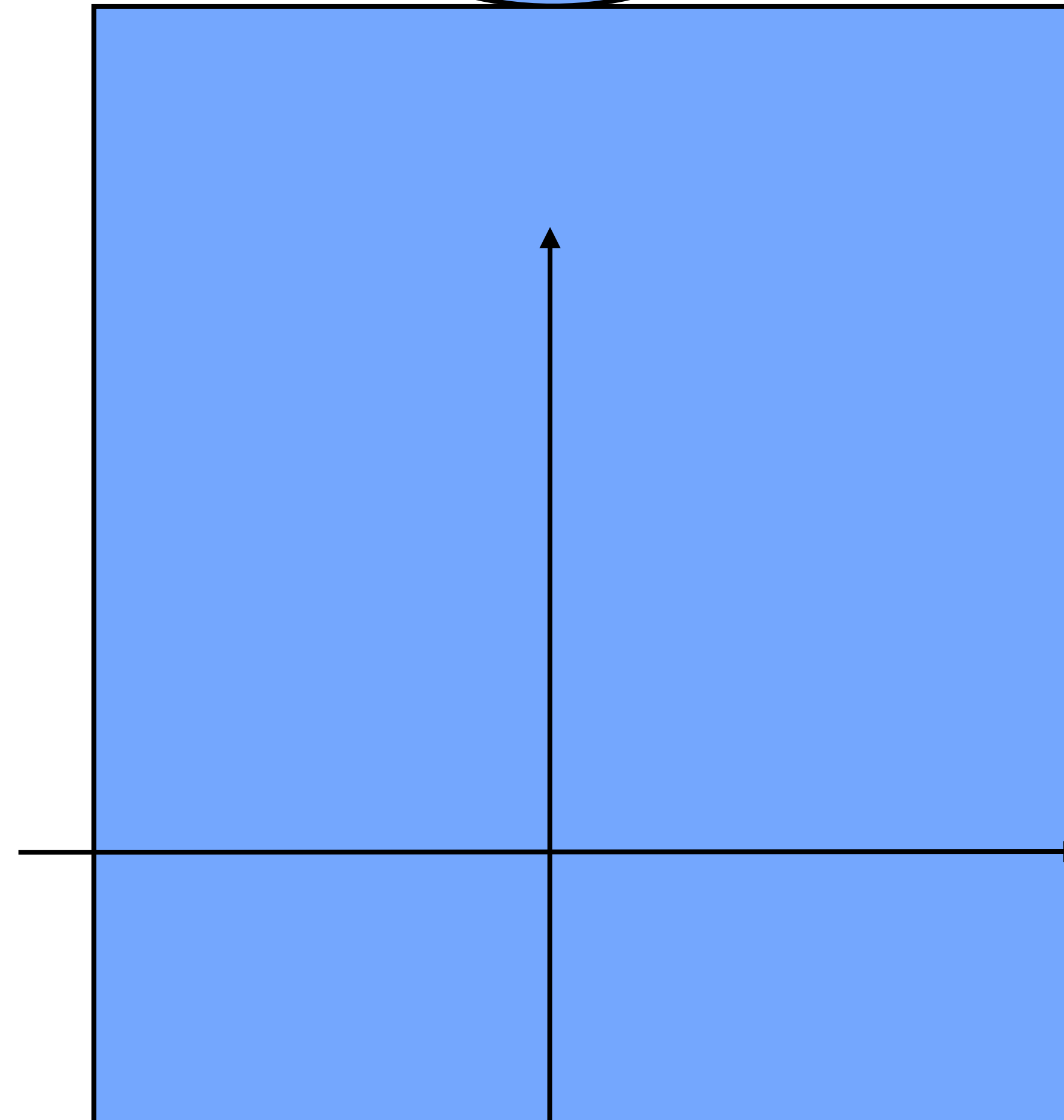
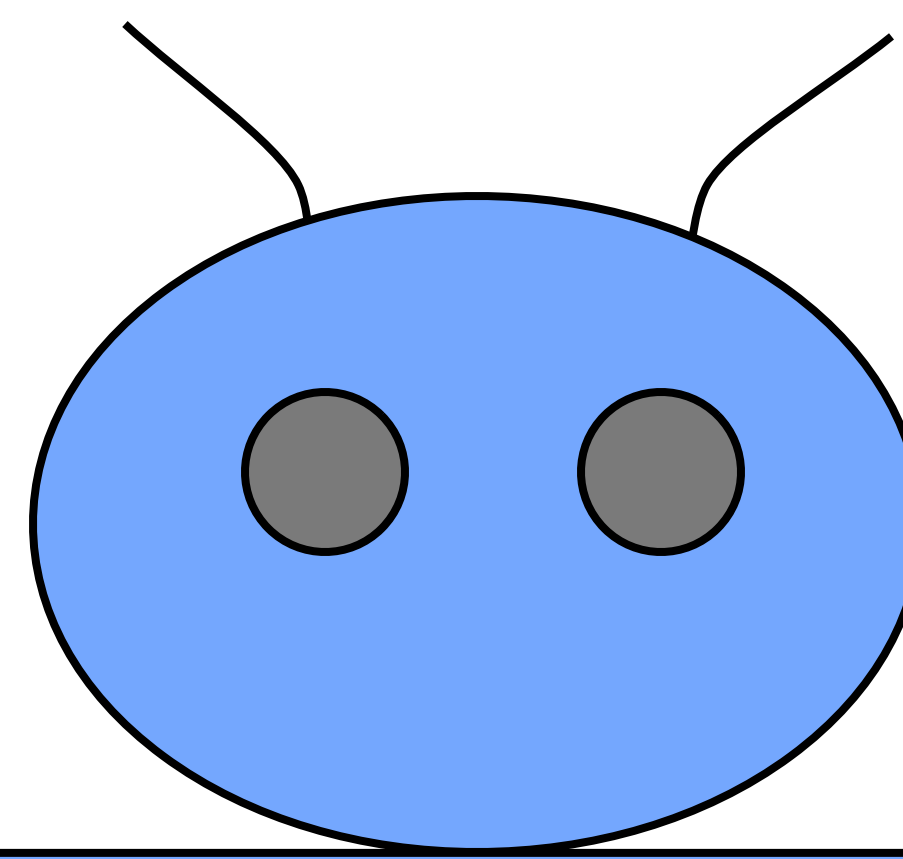
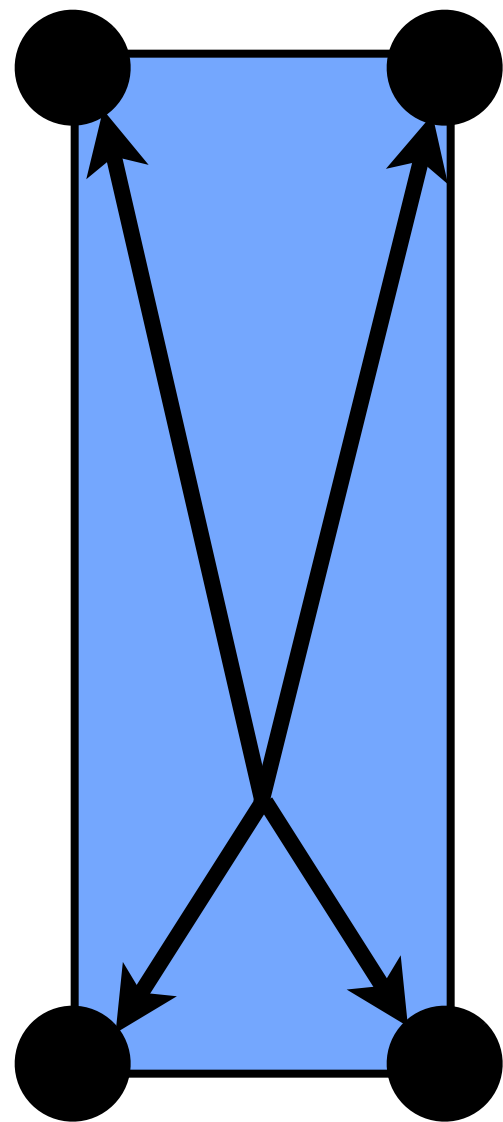




Boxy the robot



p_{link}

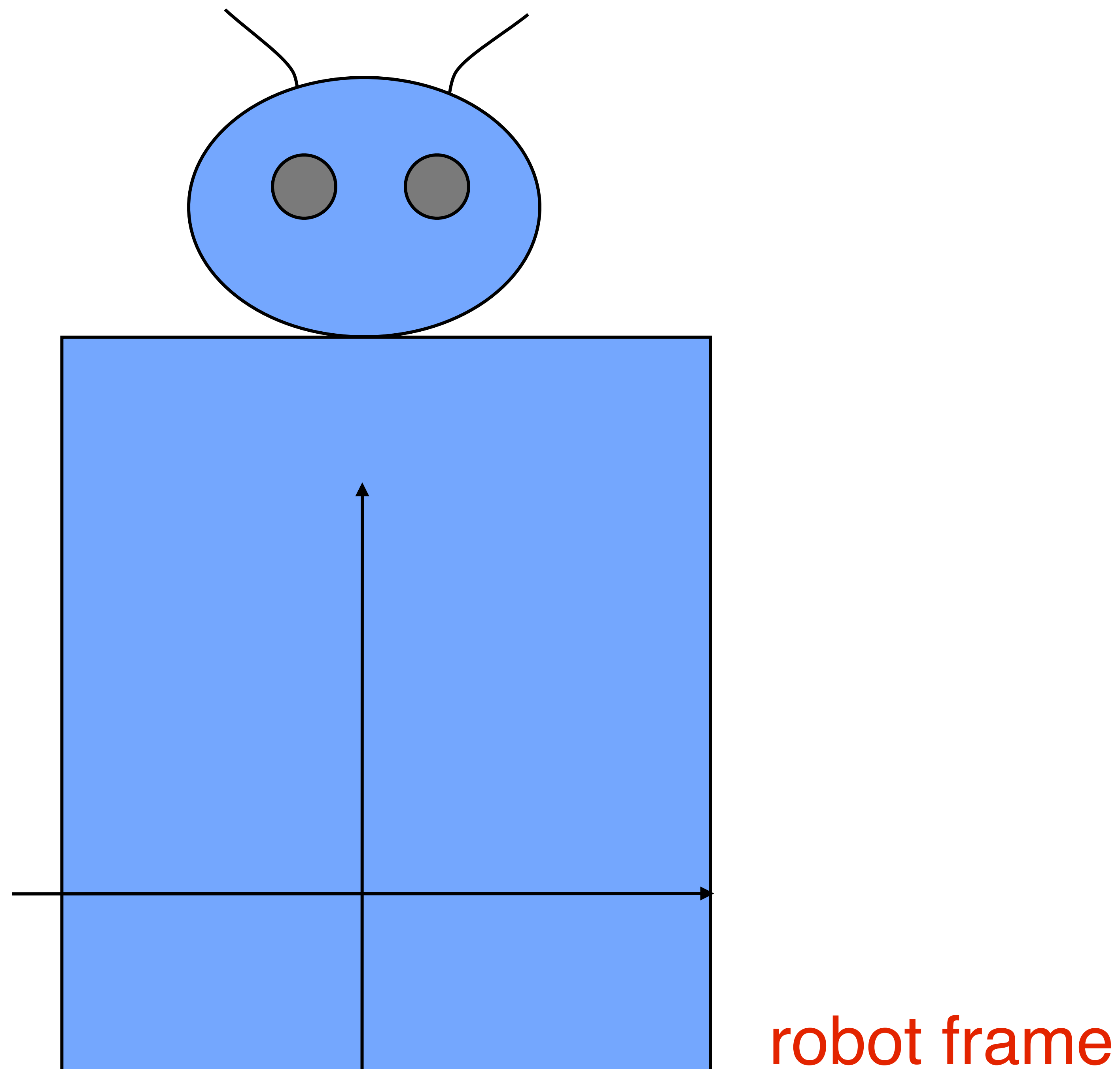
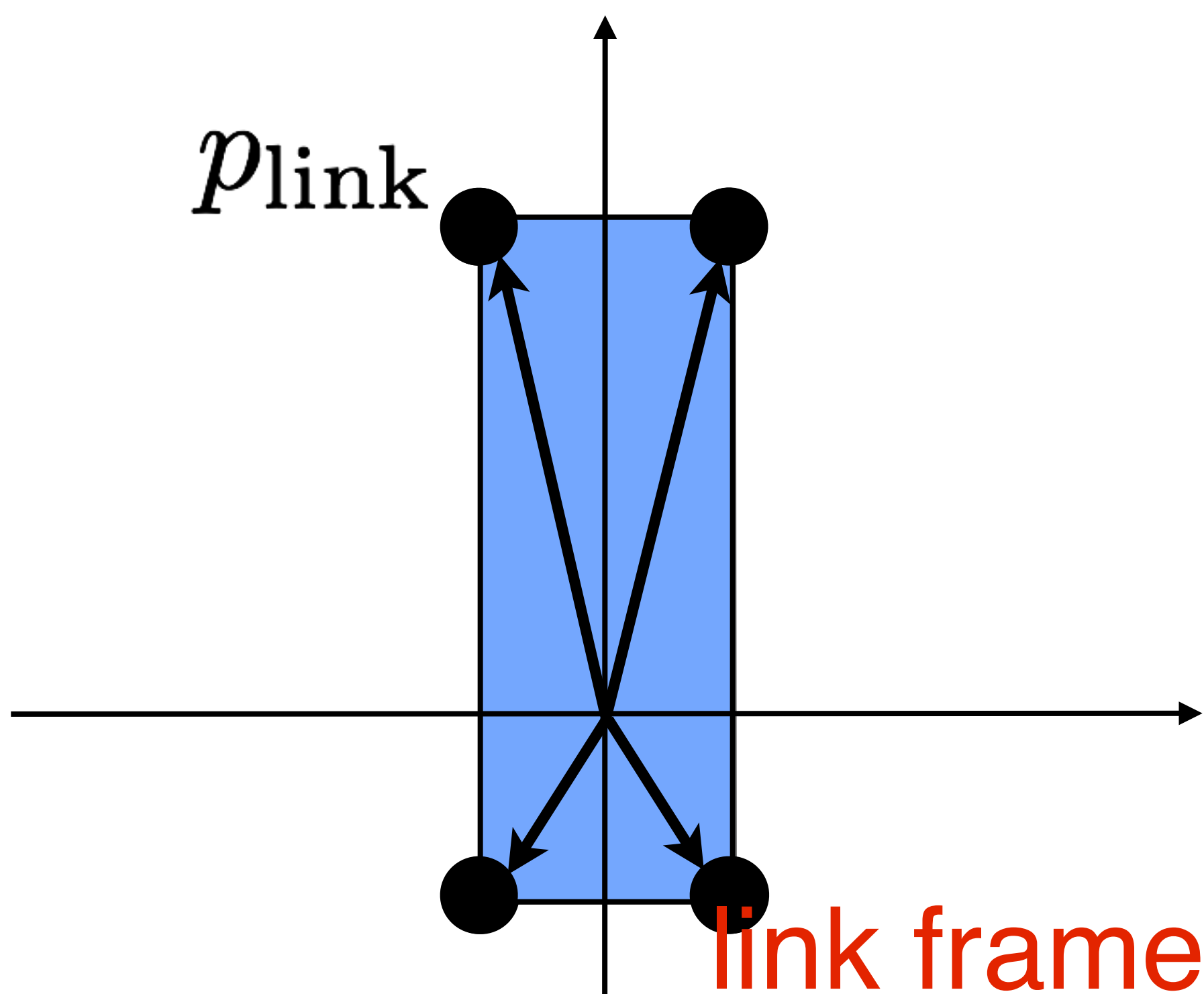


robot frame



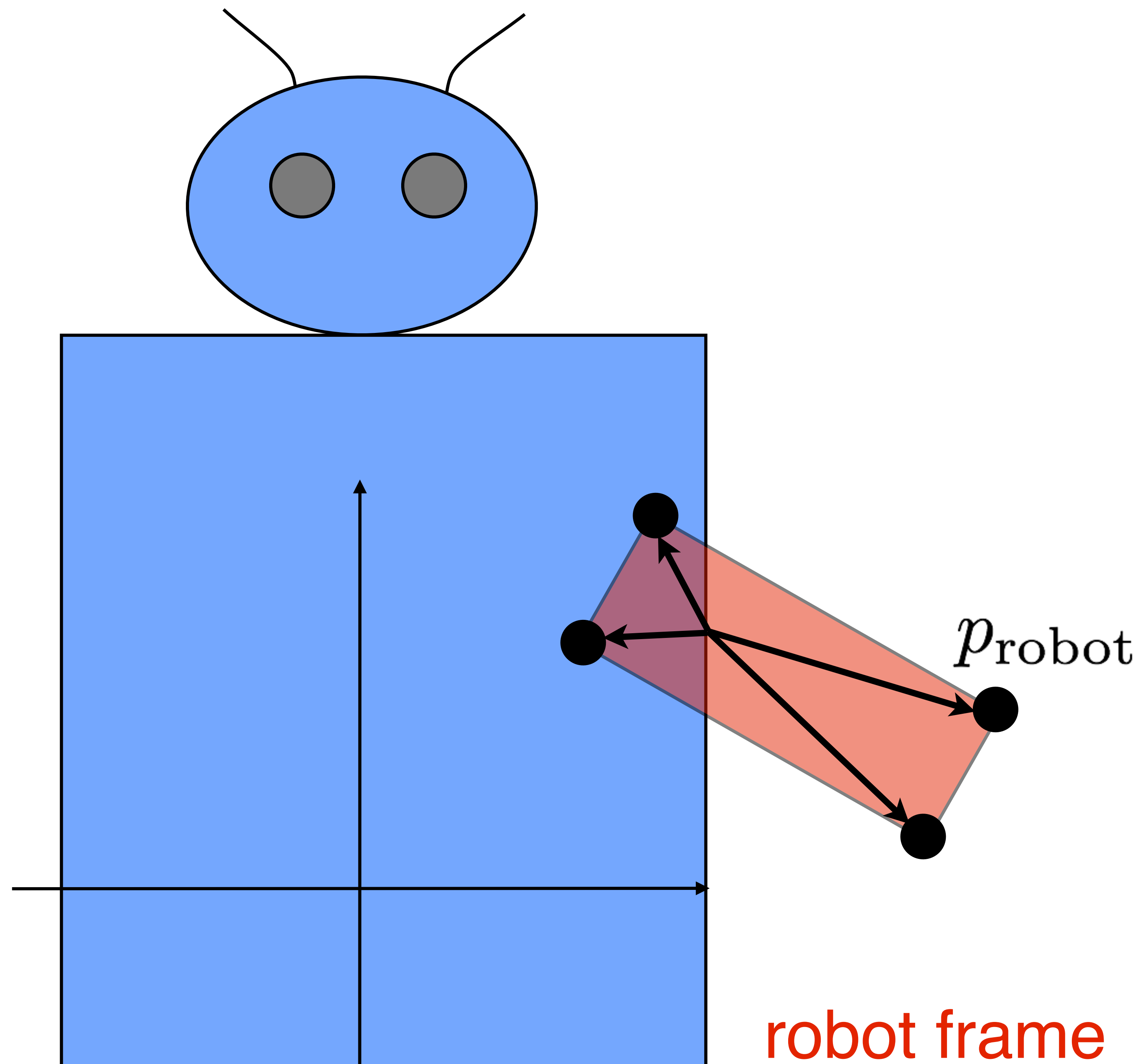
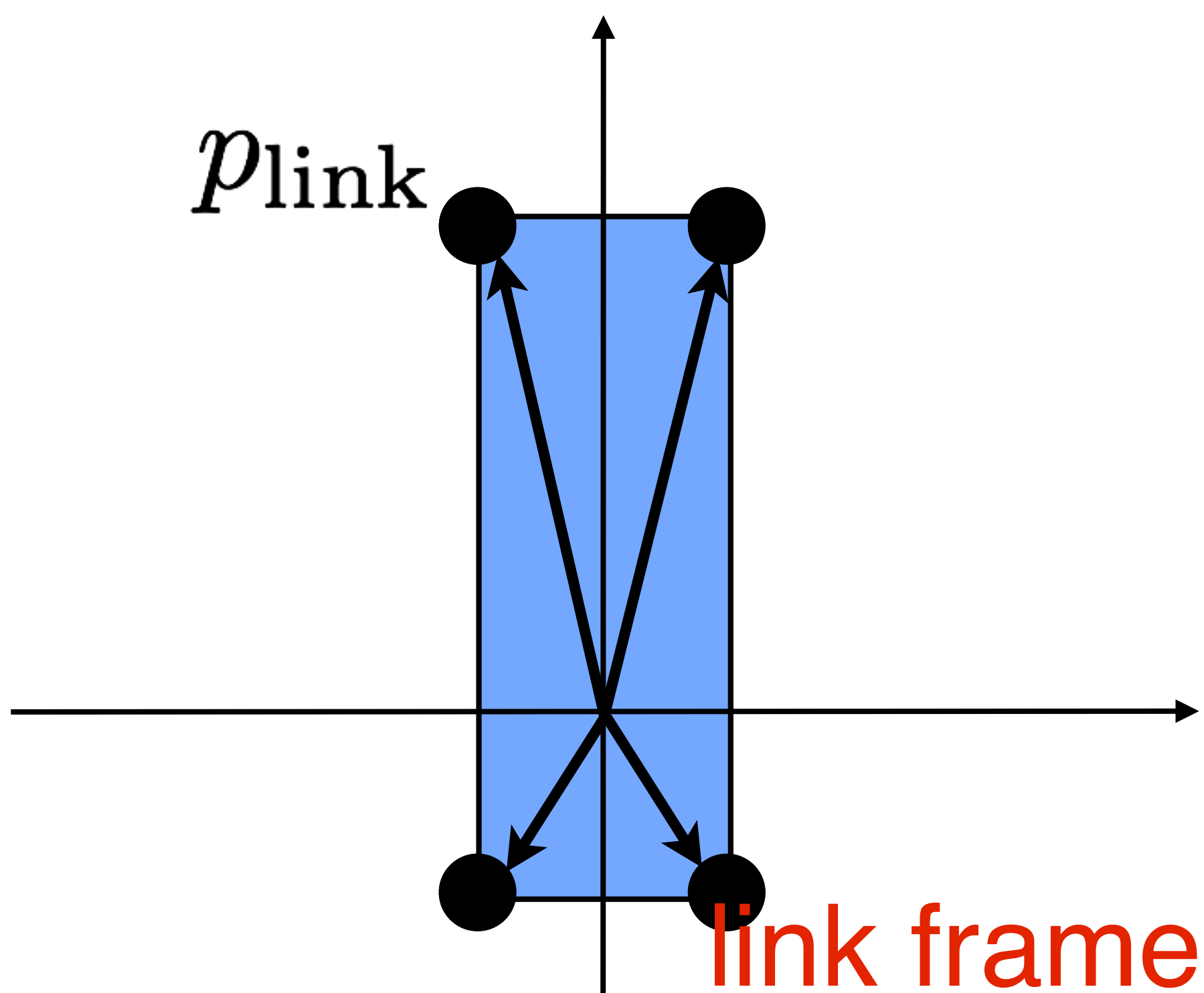
Transform the link frame and its vertices into the robot frame

$$p_{\text{robot}} = T_{\text{link}}^{\text{robot}} p_{\text{link}}$$

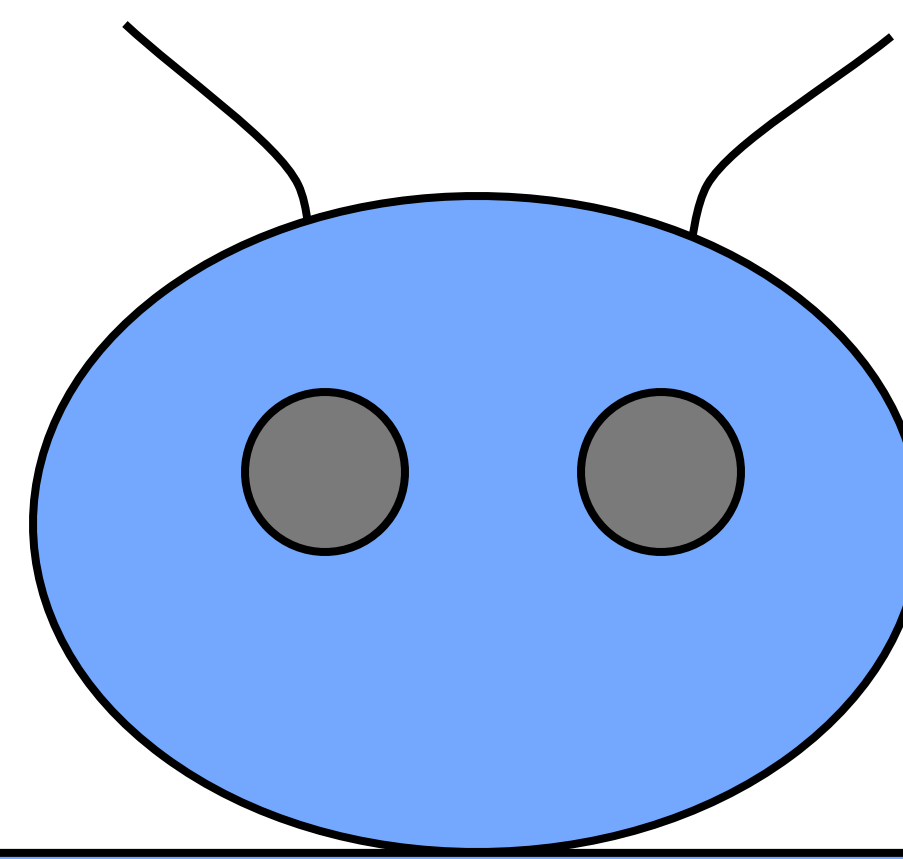


Transform the link frame and its vertices into the robot frame

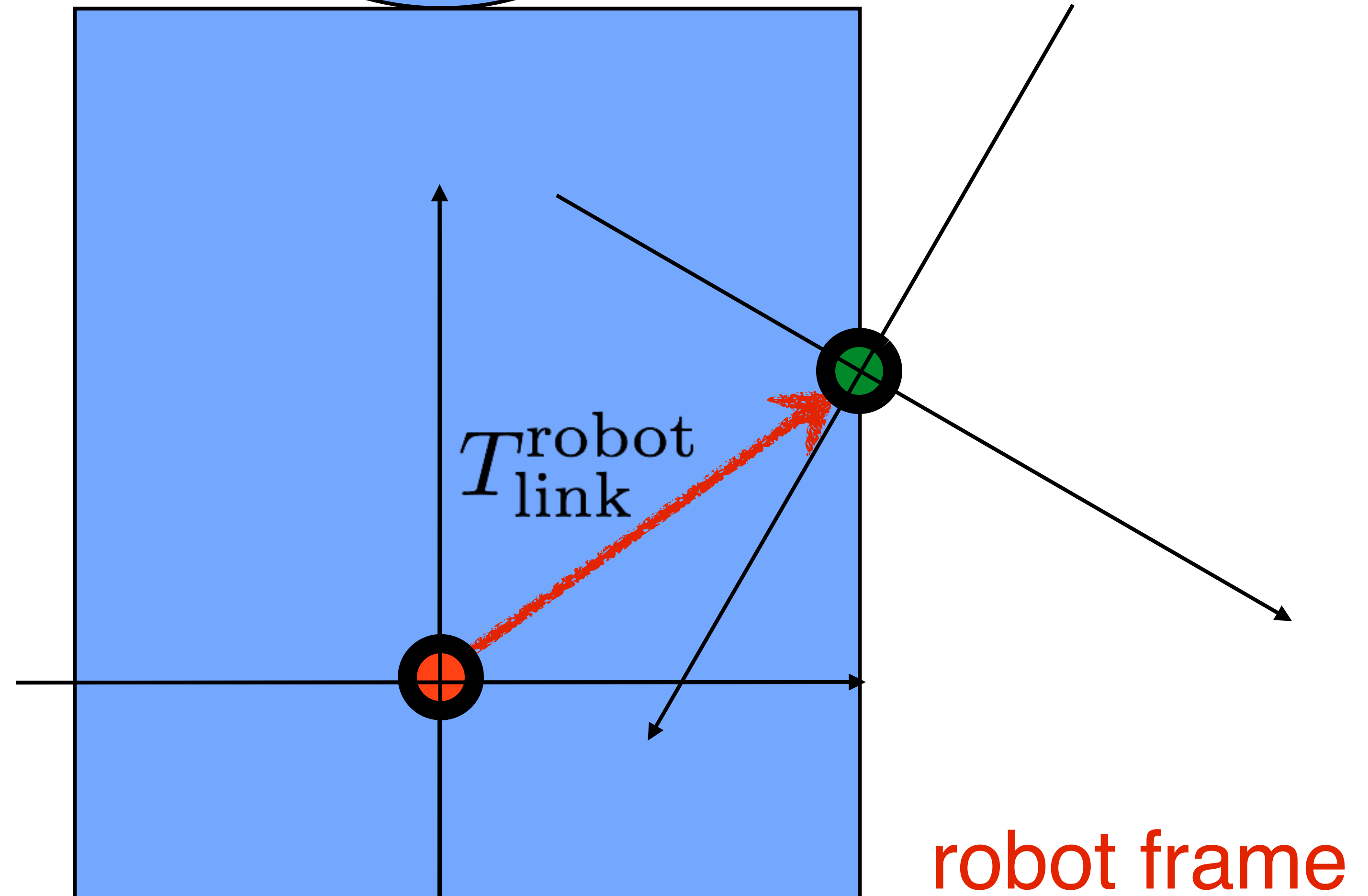
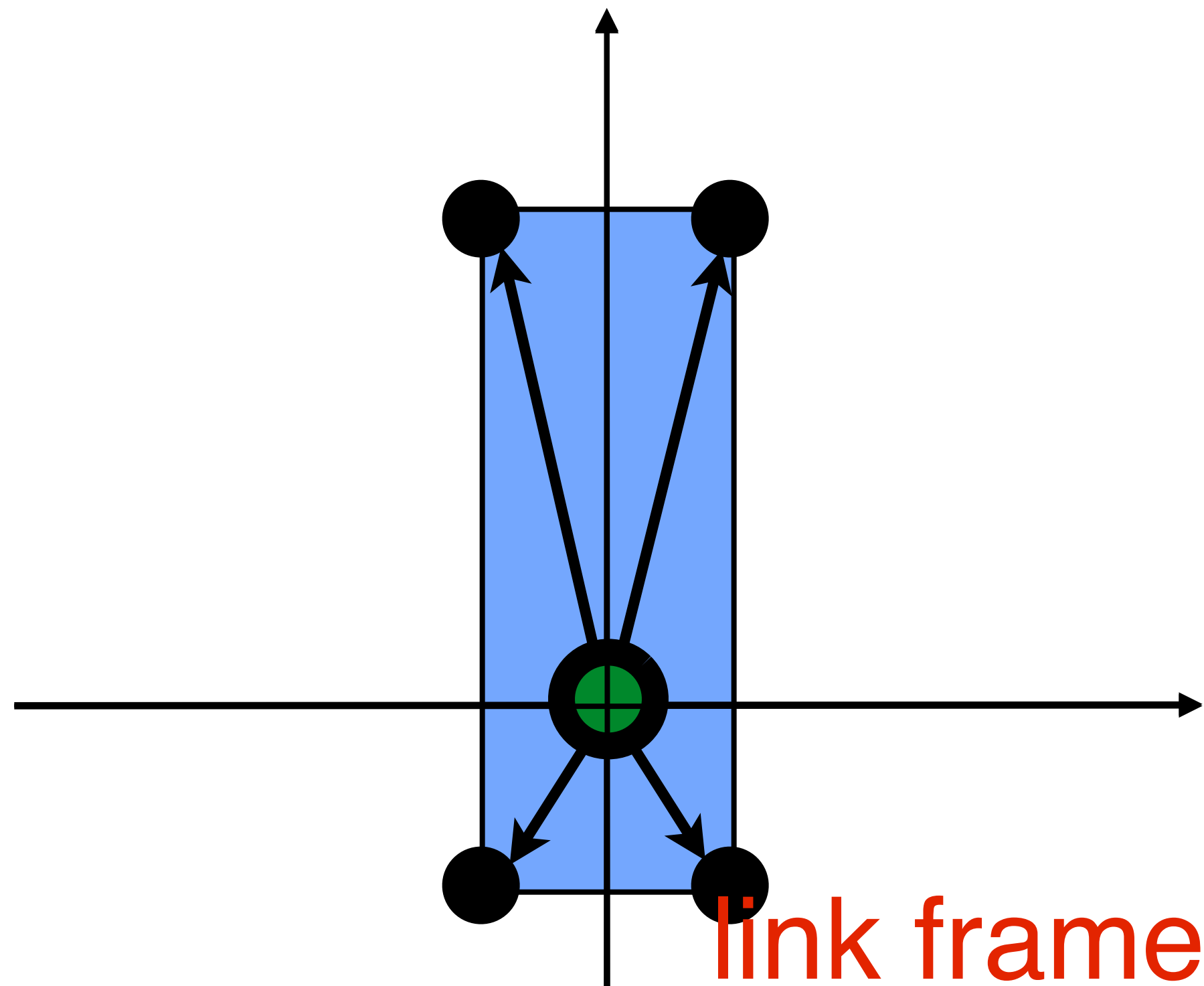
$$p_{\text{robot}} = T_{\text{link}}^{\text{robot}} p_{\text{link}}$$



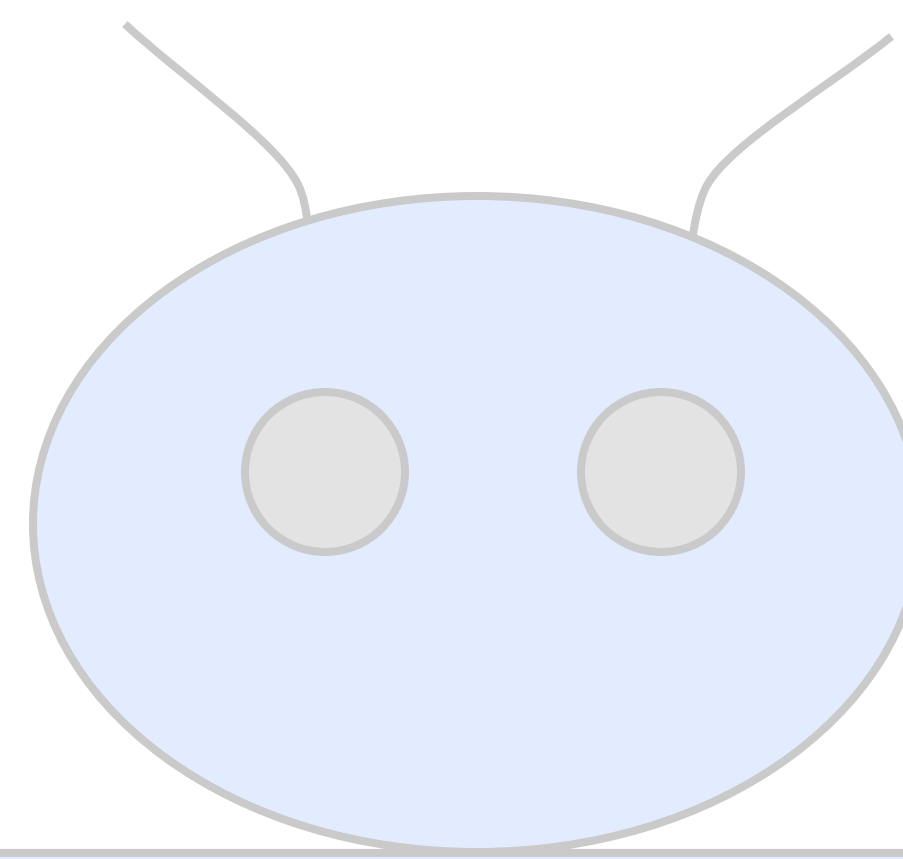
$$T_{\text{link}}^{\text{robot}} = \begin{bmatrix} R_{00} & R_{01} & d_x \\ R_{10} & R_{11} & d_y \\ 0 & 0 & 1 \end{bmatrix}$$



Can we think about this frame relation in steps?

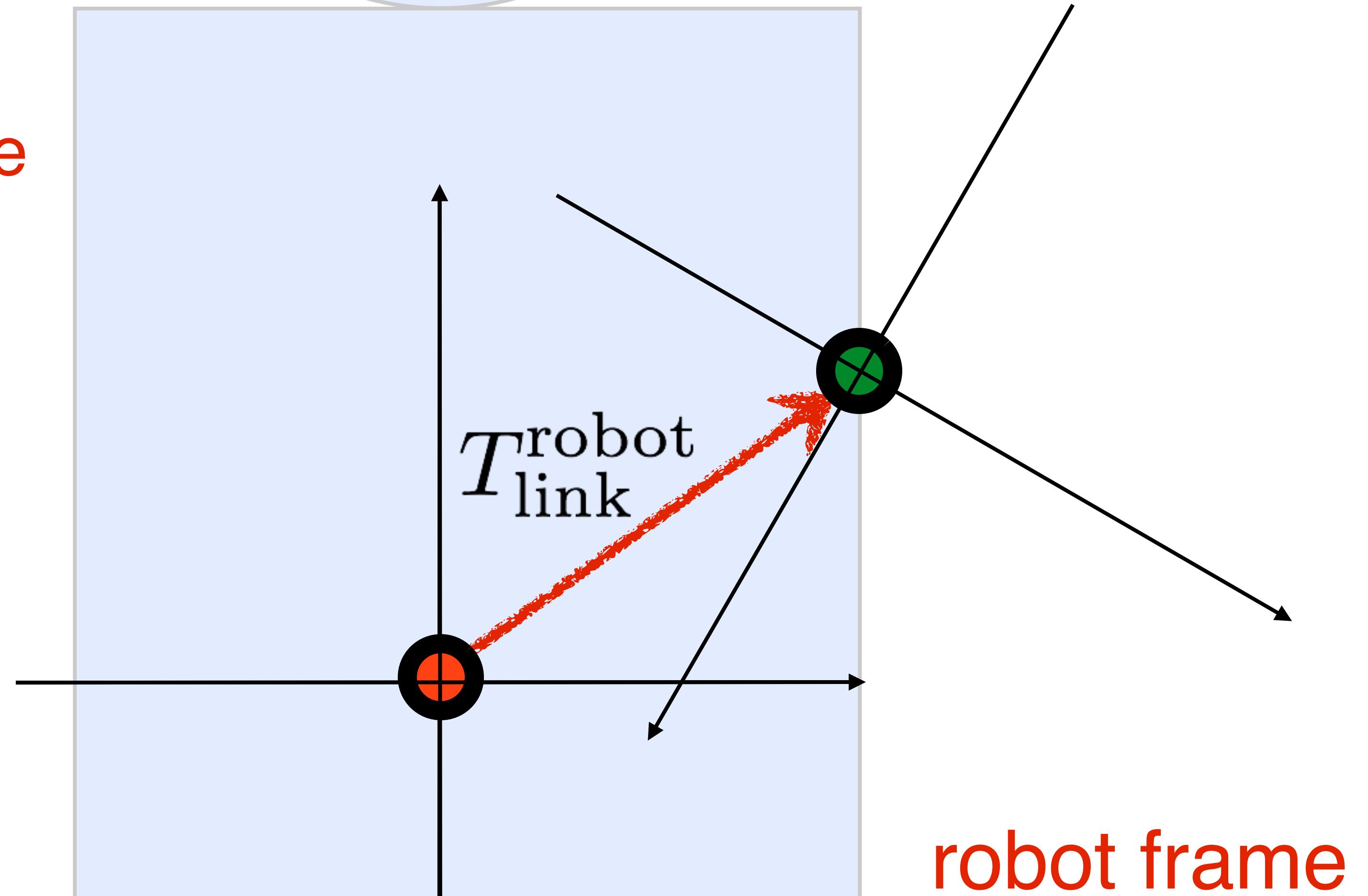
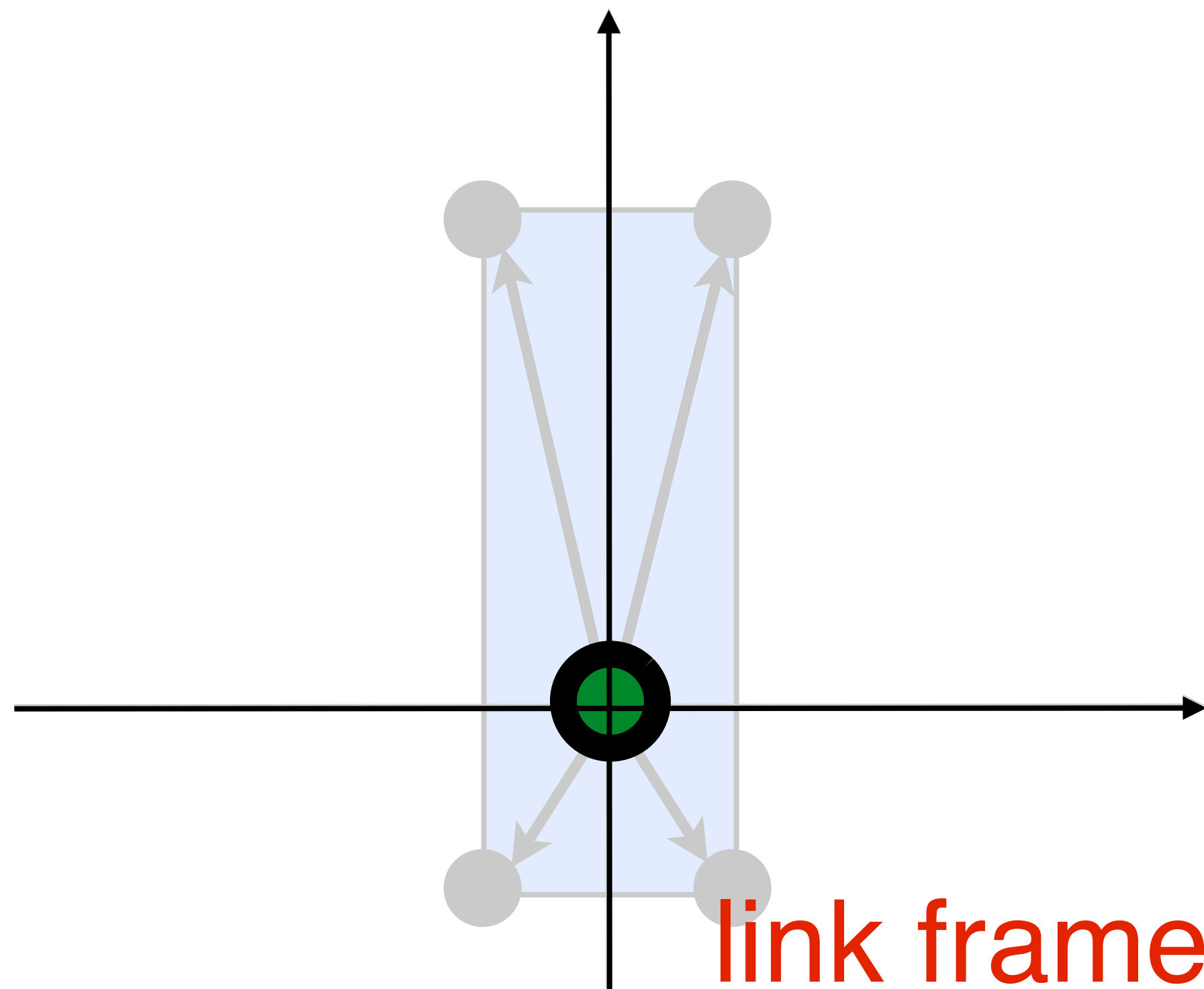


$$T_{\text{link}}^{\text{robot}} = \begin{bmatrix} R_{00} & R_{01} & d_x \\ R_{10} & R_{11} & d_y \\ 0 & 0 & 1 \end{bmatrix}$$

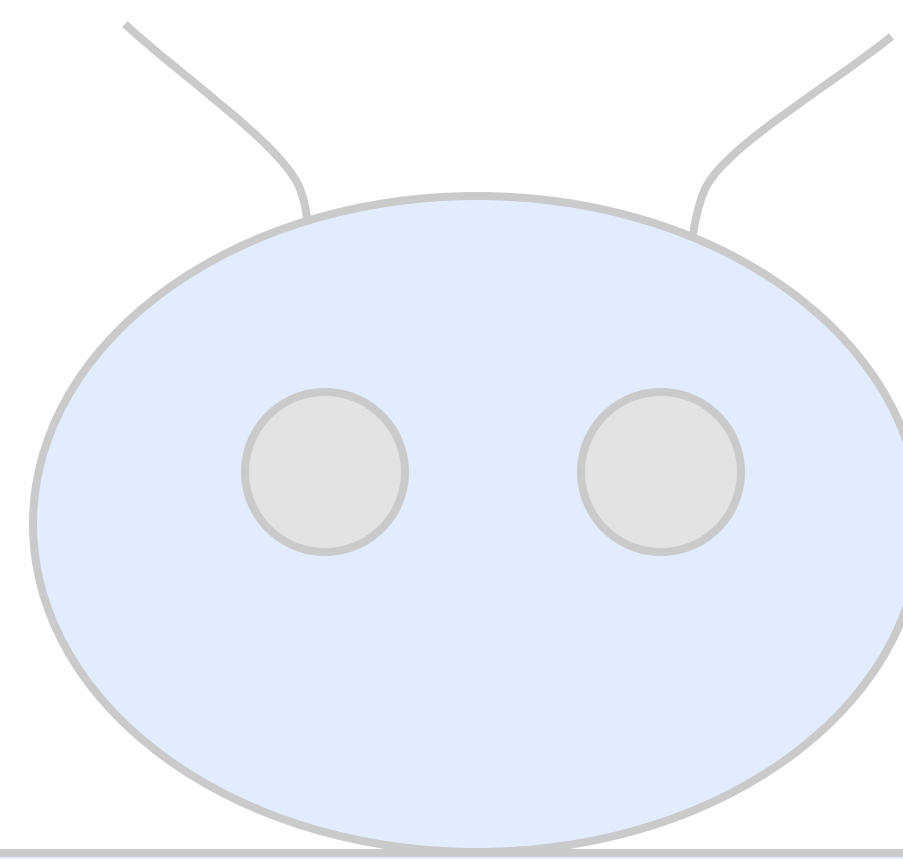


Transformed frame
for link wrt. robot

First consider link in its own frame

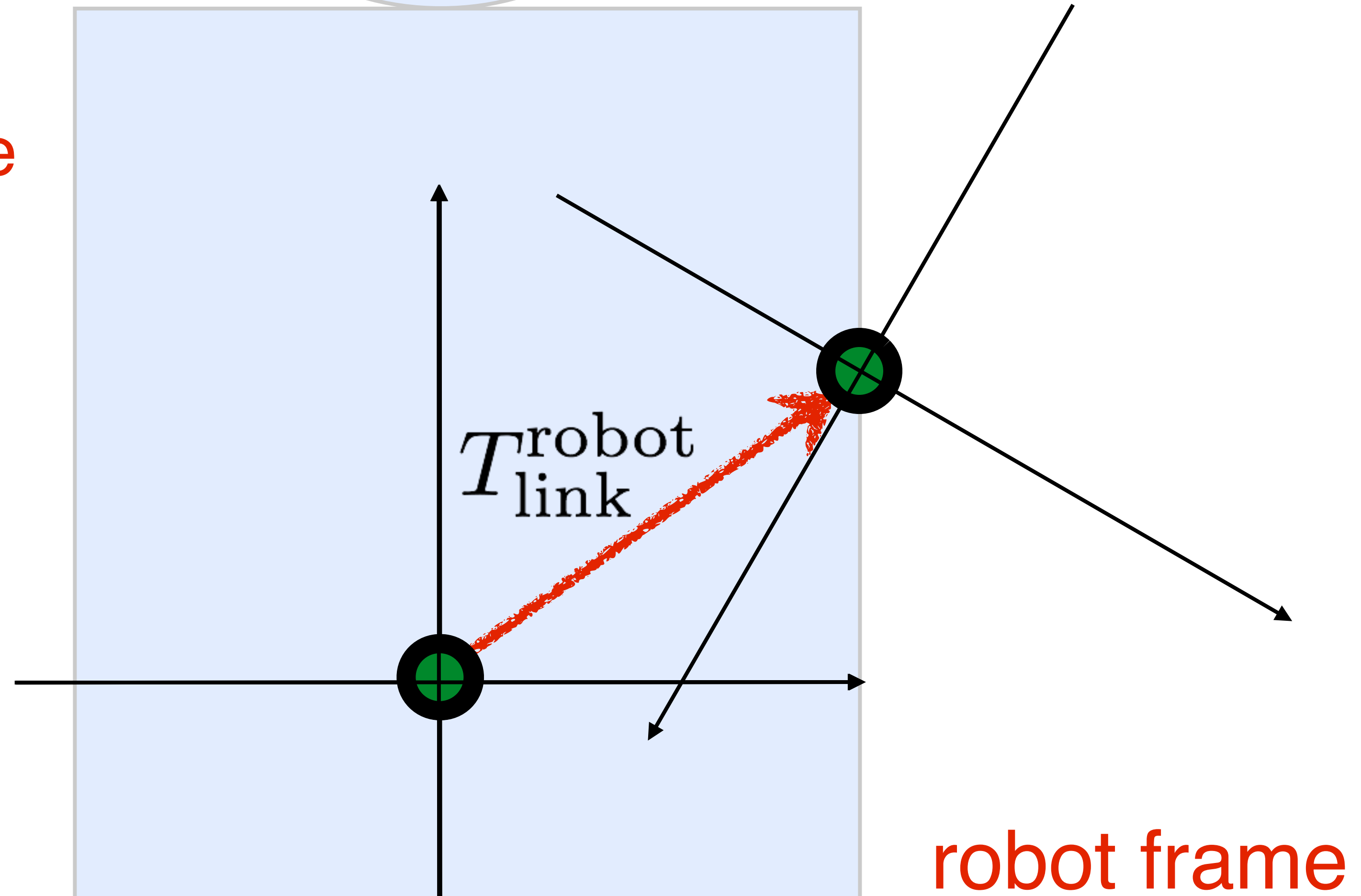
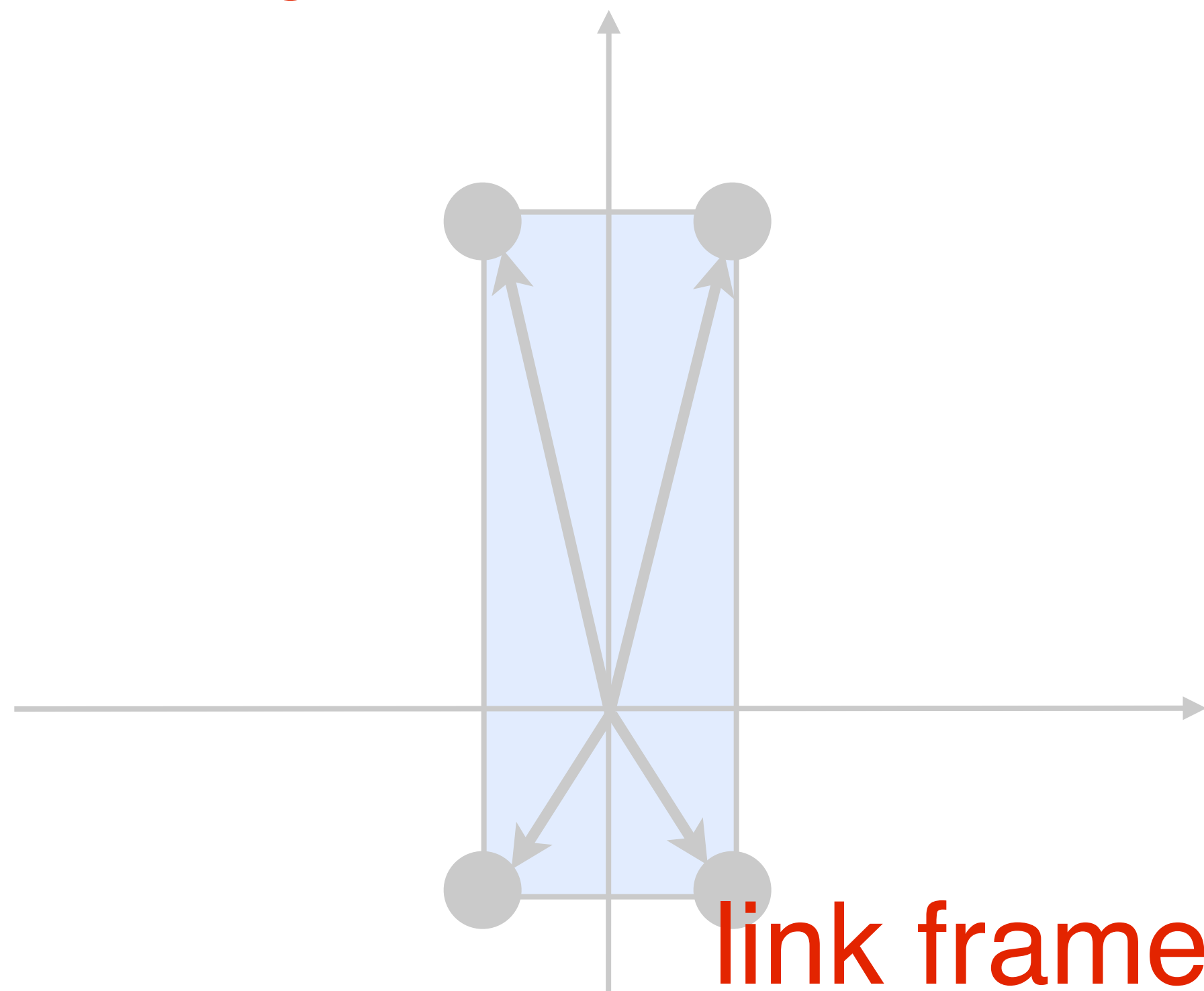


$$T_{\text{link}}^{\text{robot}} = \begin{bmatrix} R_{00} & R_{01} & d_x \\ R_{10} & R_{11} & d_y \\ 0 & 0 & 1 \end{bmatrix}$$

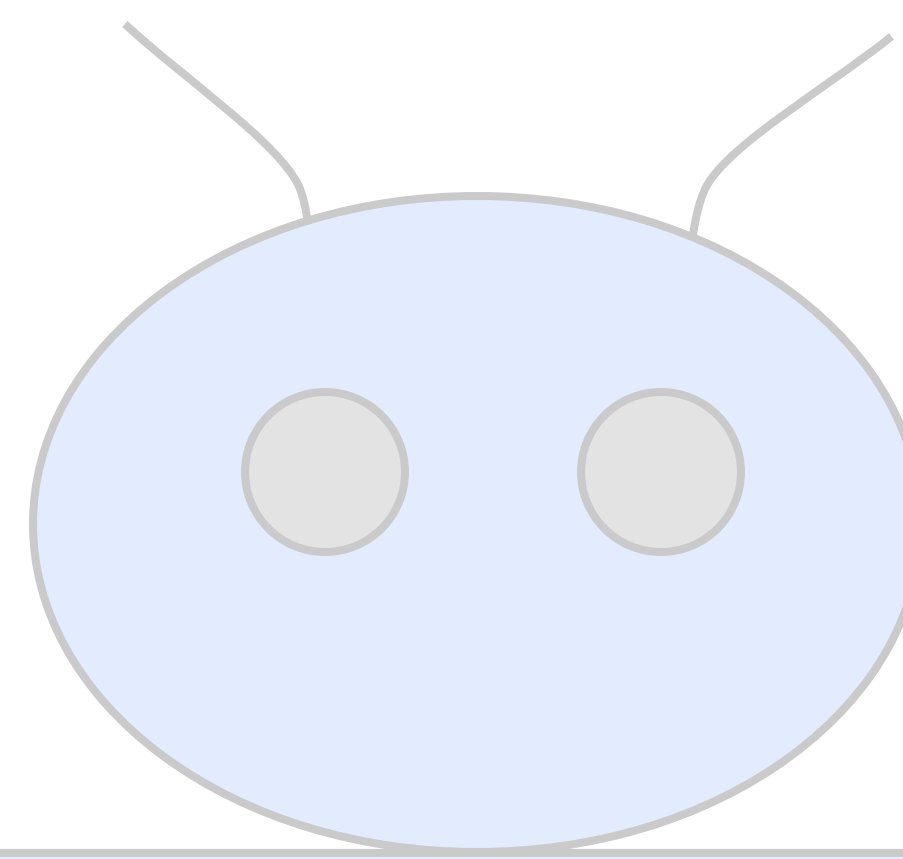


Transformed frame
for link wrt. robot

as aligned with robot base frame

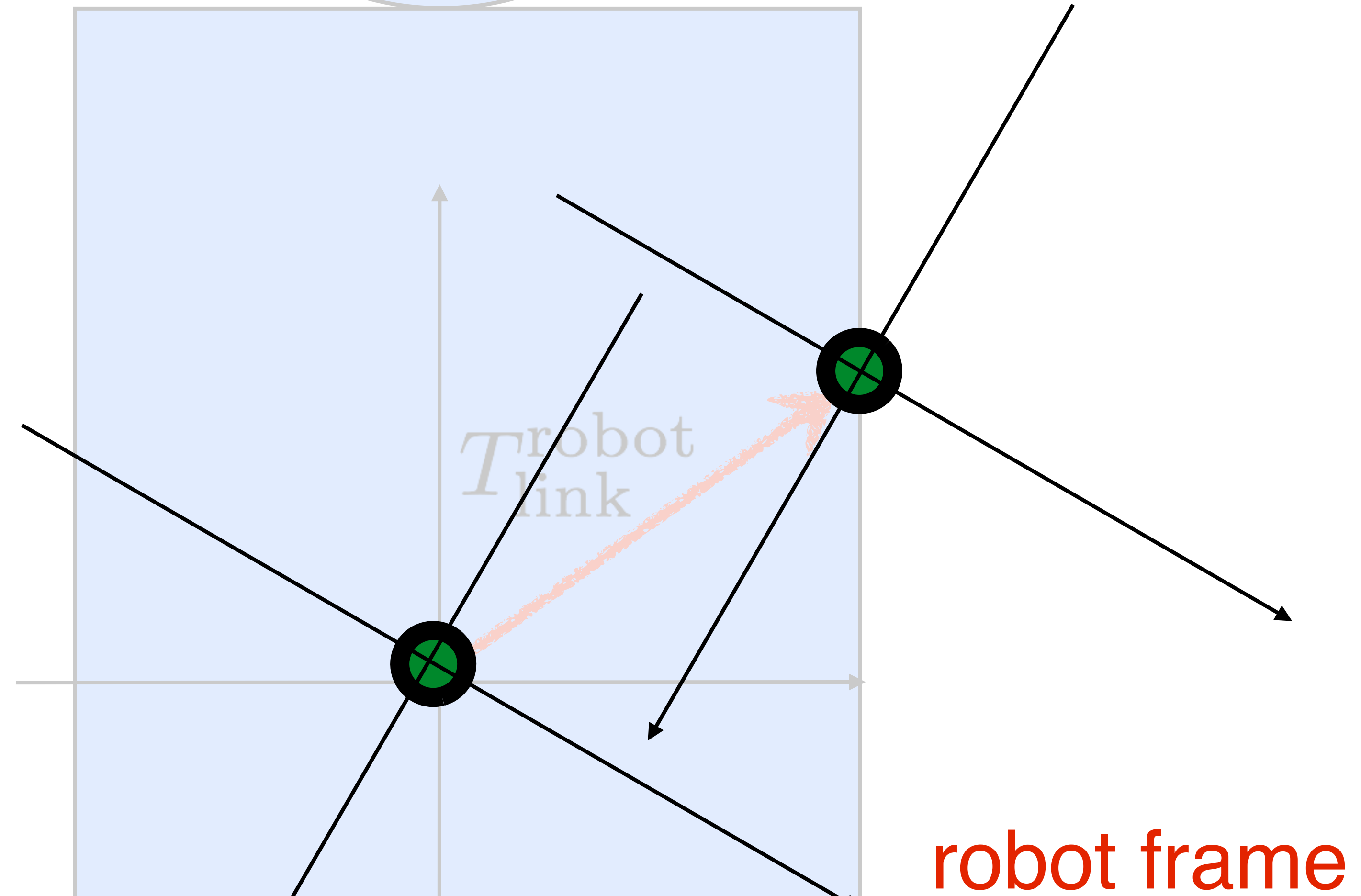
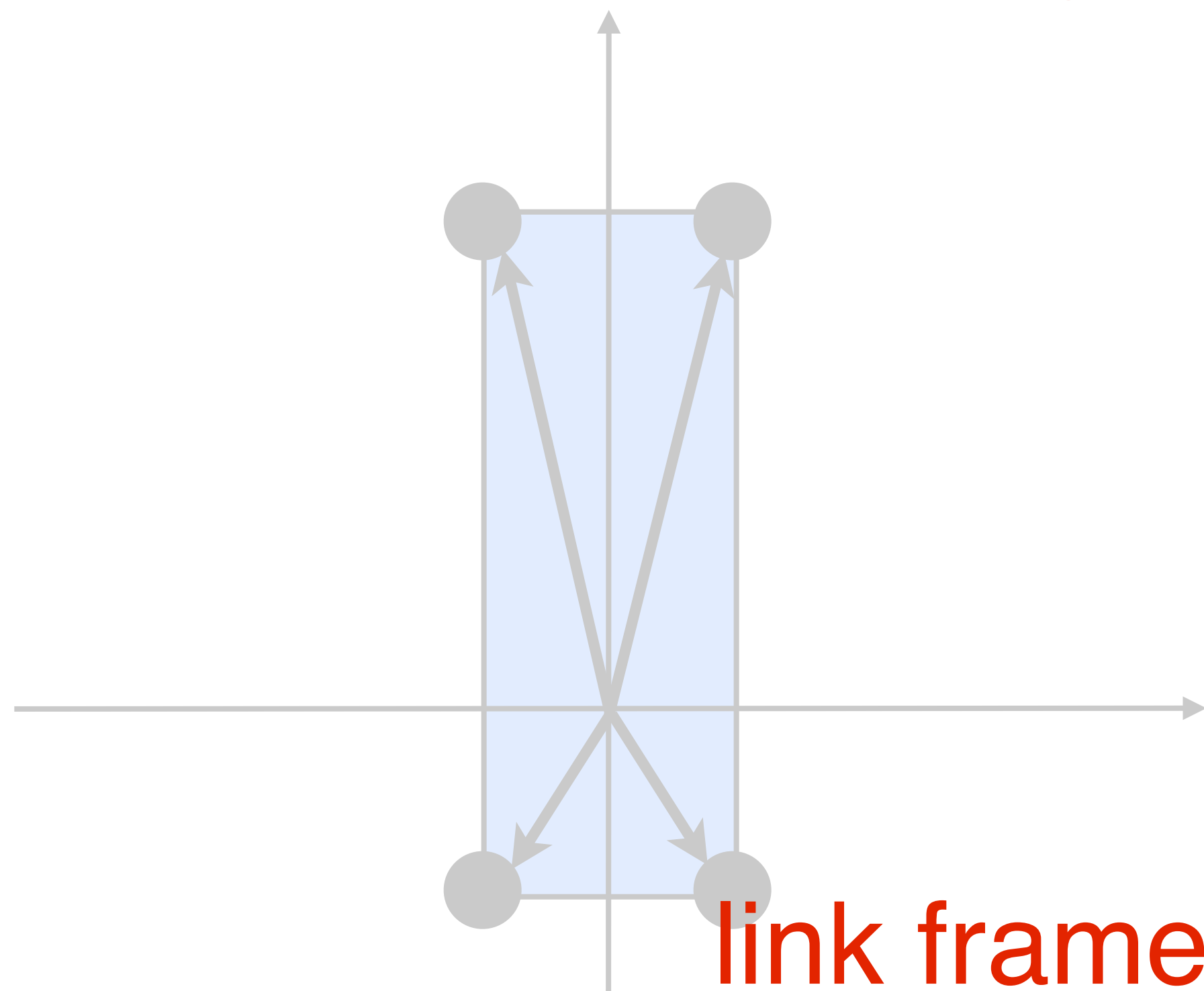


$$T_{\text{link}}^{\text{robot}} = \begin{bmatrix} R_{00} & R_{01} & d_x \\ R_{10} & R_{11} & d_y \\ 0 & 0 & 1 \end{bmatrix}$$

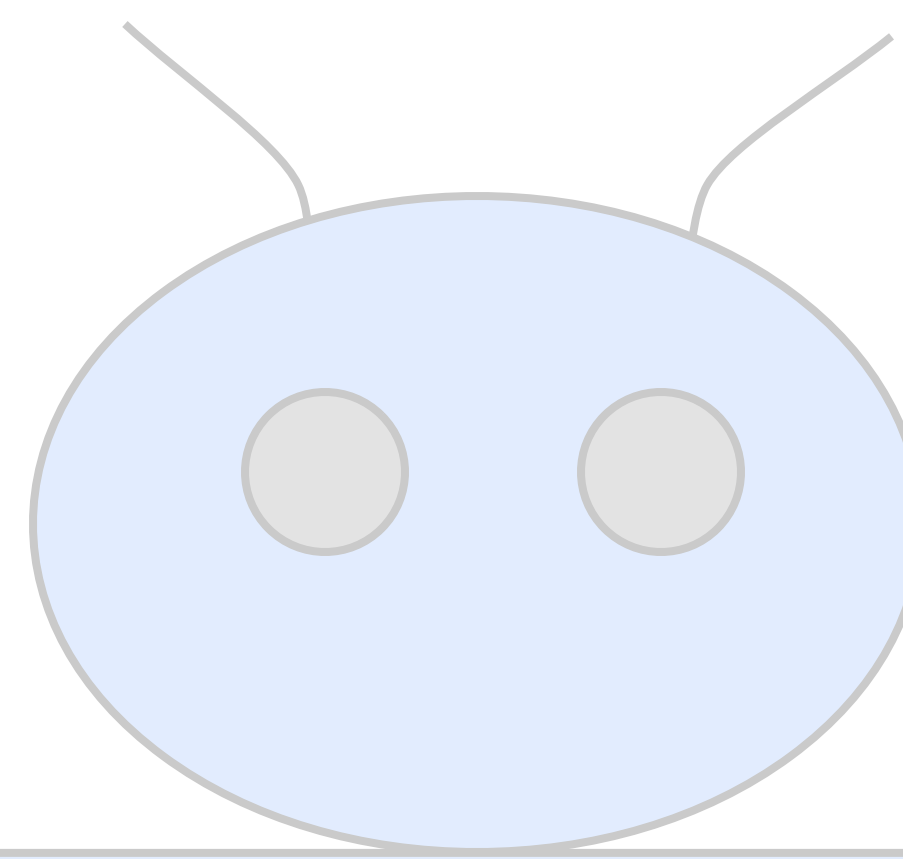


Transformed frame
for link wrt. robot

Rotate link frame by R

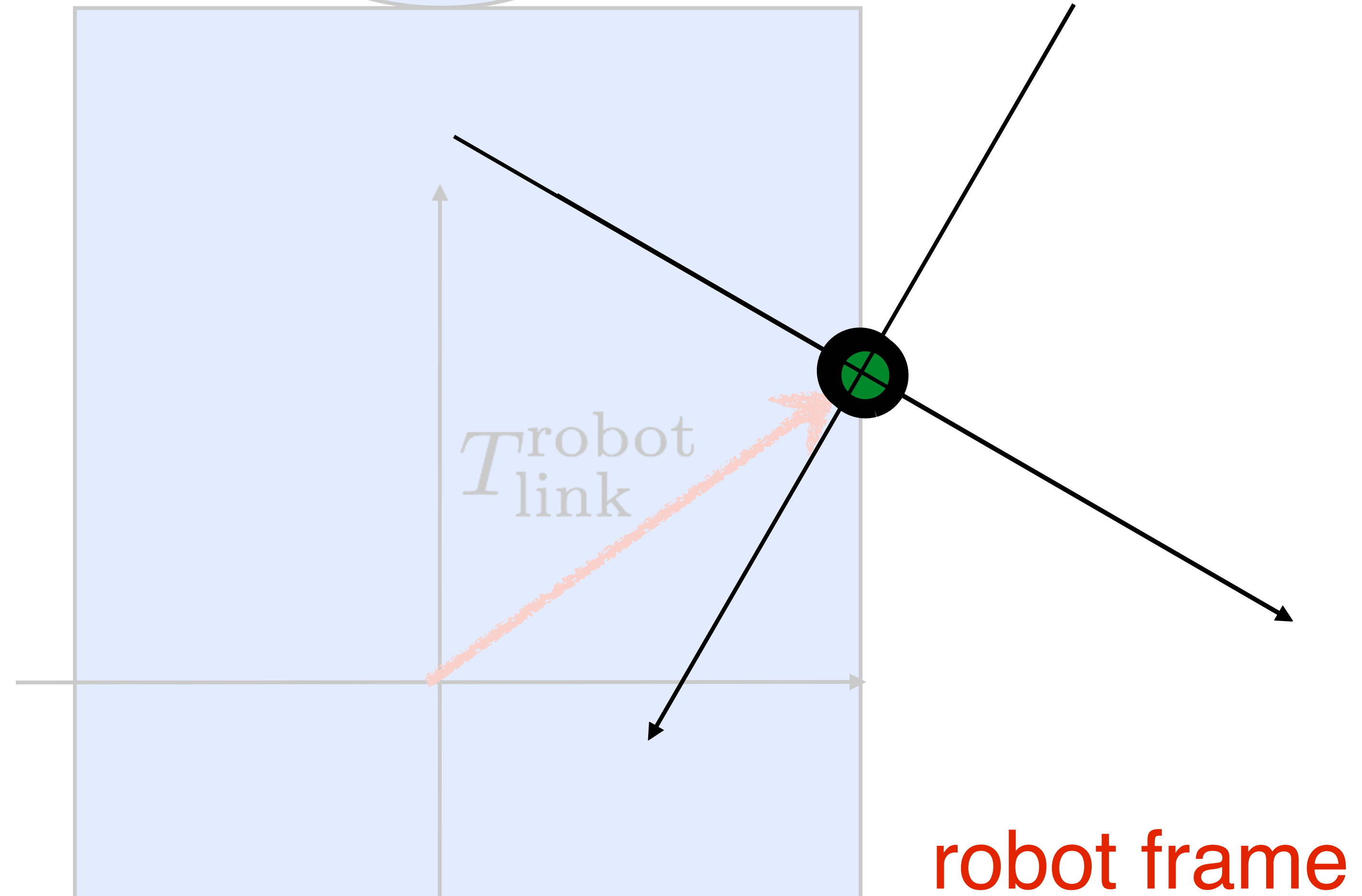
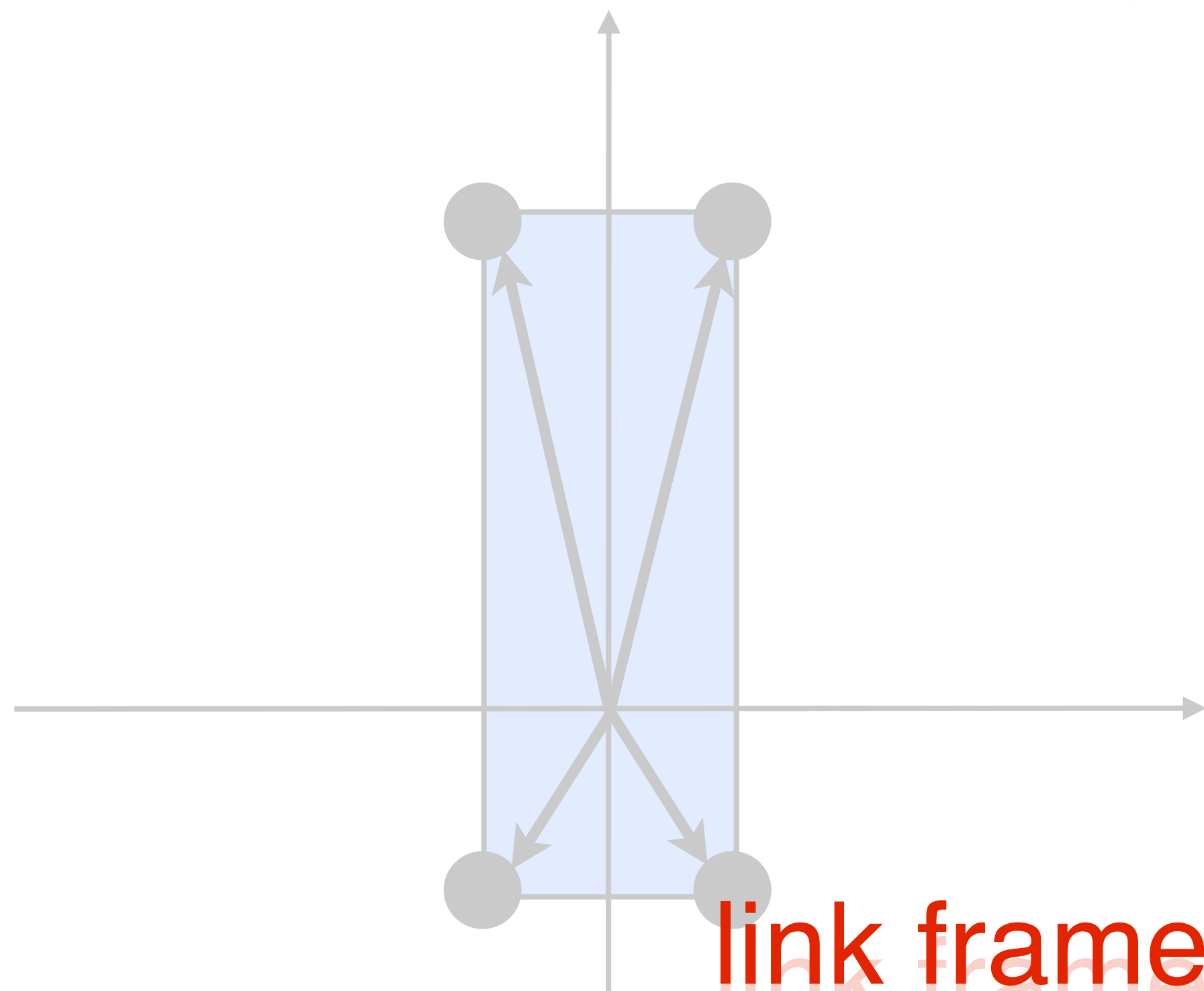


$$T_{\text{link}}^{\text{robot}} = \begin{bmatrix} R_{00} & R_{01} & d_x \\ R_{10} & R_{11} & d_y \\ 0 & 0 & 1 \end{bmatrix}$$



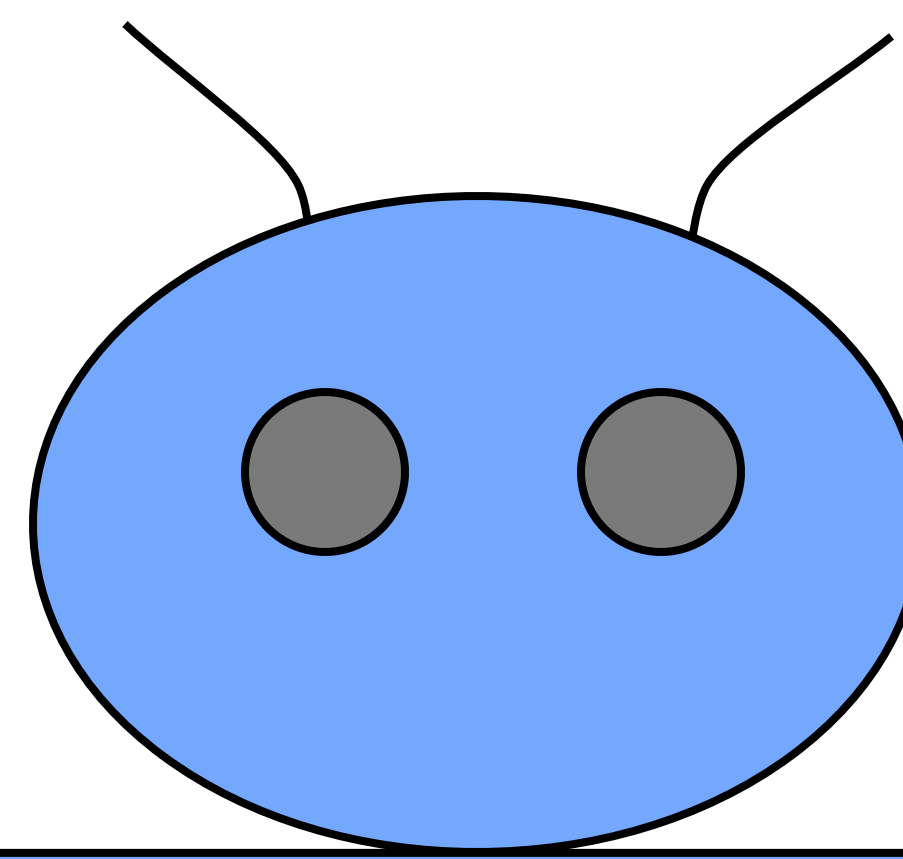
Transformed frame
for link wrt. robot

Translate link frame by d

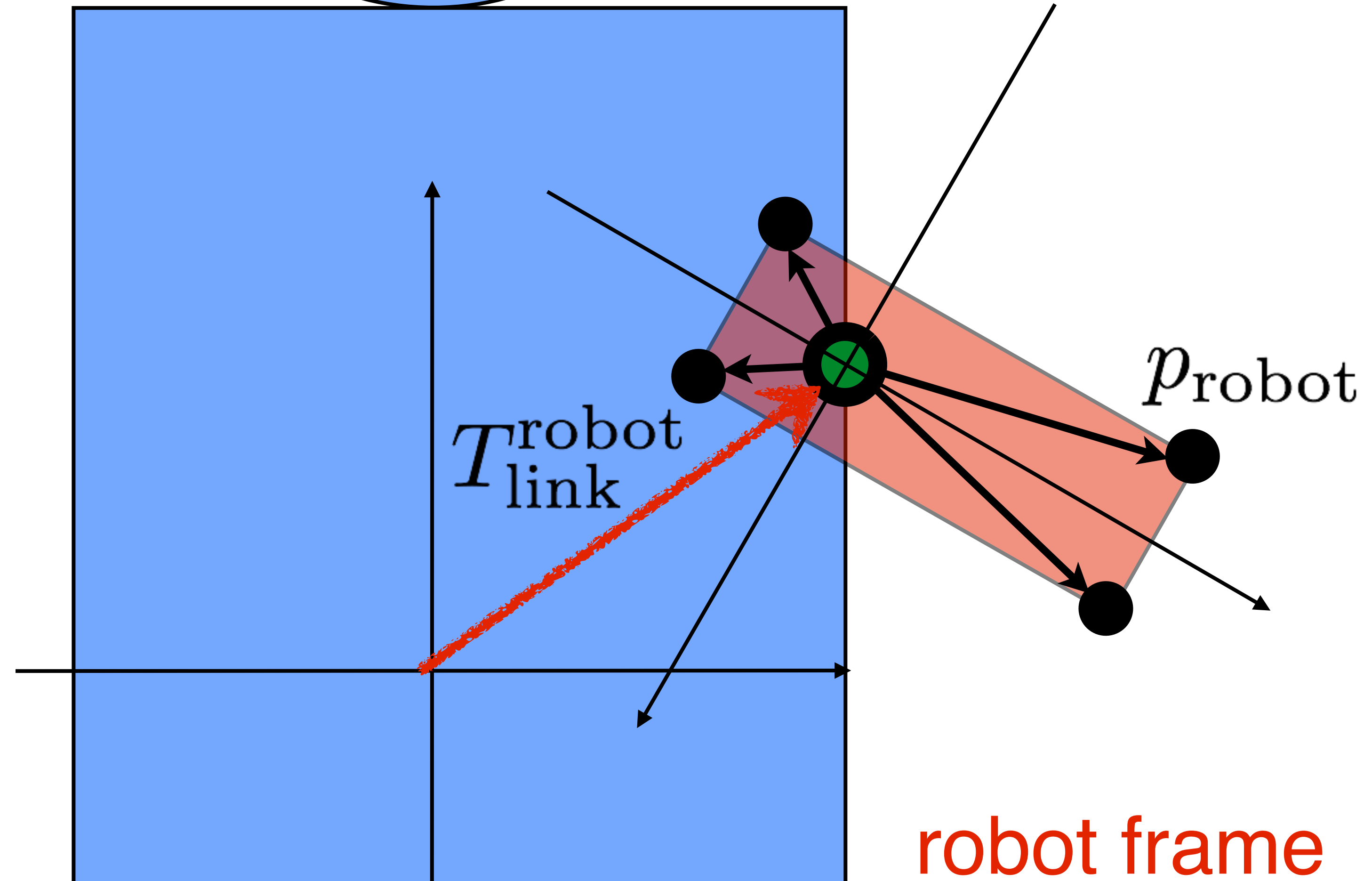
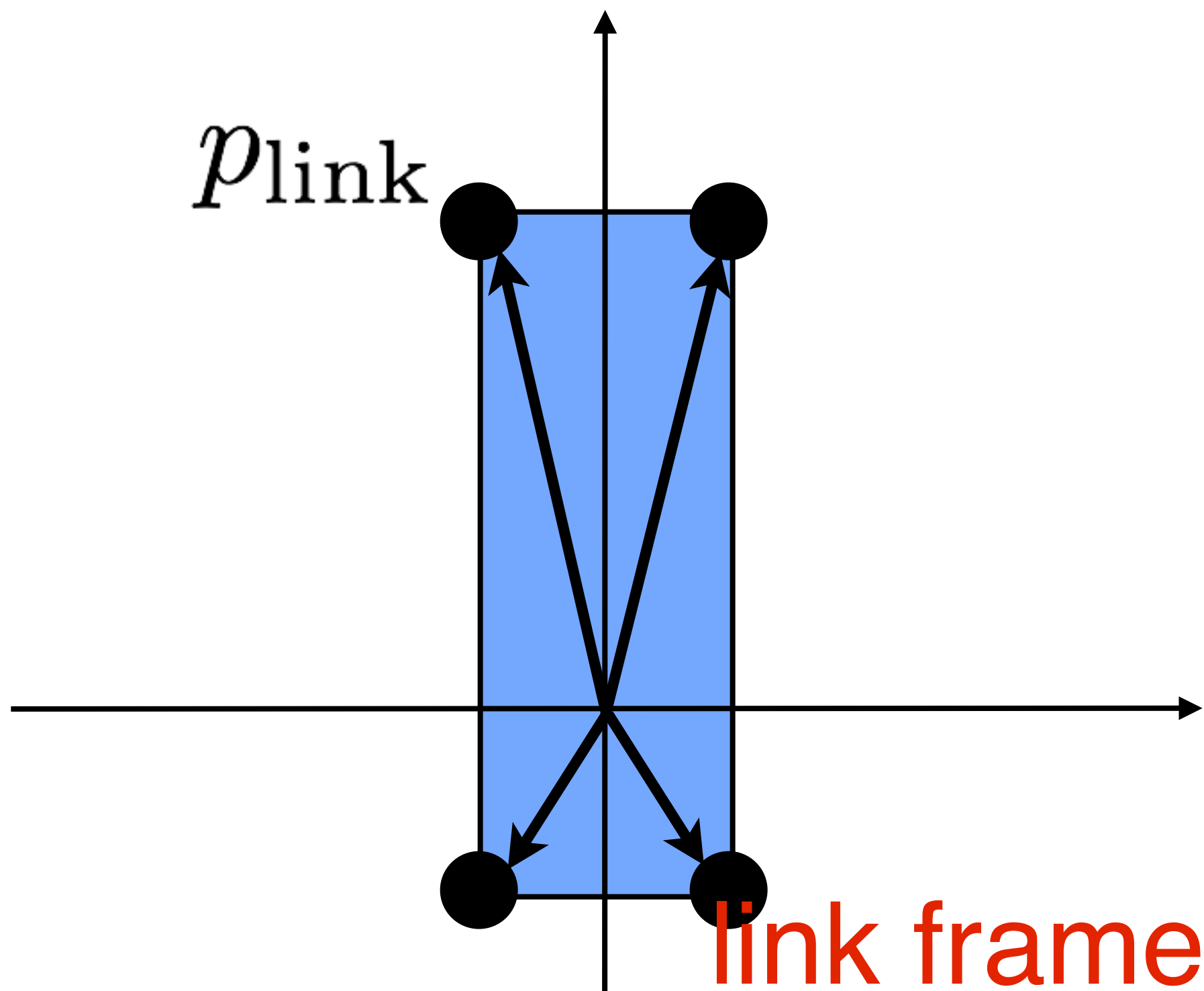


$$T_{\text{link}}^{\text{robot}} = \begin{bmatrix} R_{00} & R_{01} & d_x \\ R_{10} & R_{11} & d_y \\ 0 & 0 & 1 \end{bmatrix}$$

$$p_{\text{robot}} = T_{\text{link}}^{\text{robot}} p_{\text{link}}$$

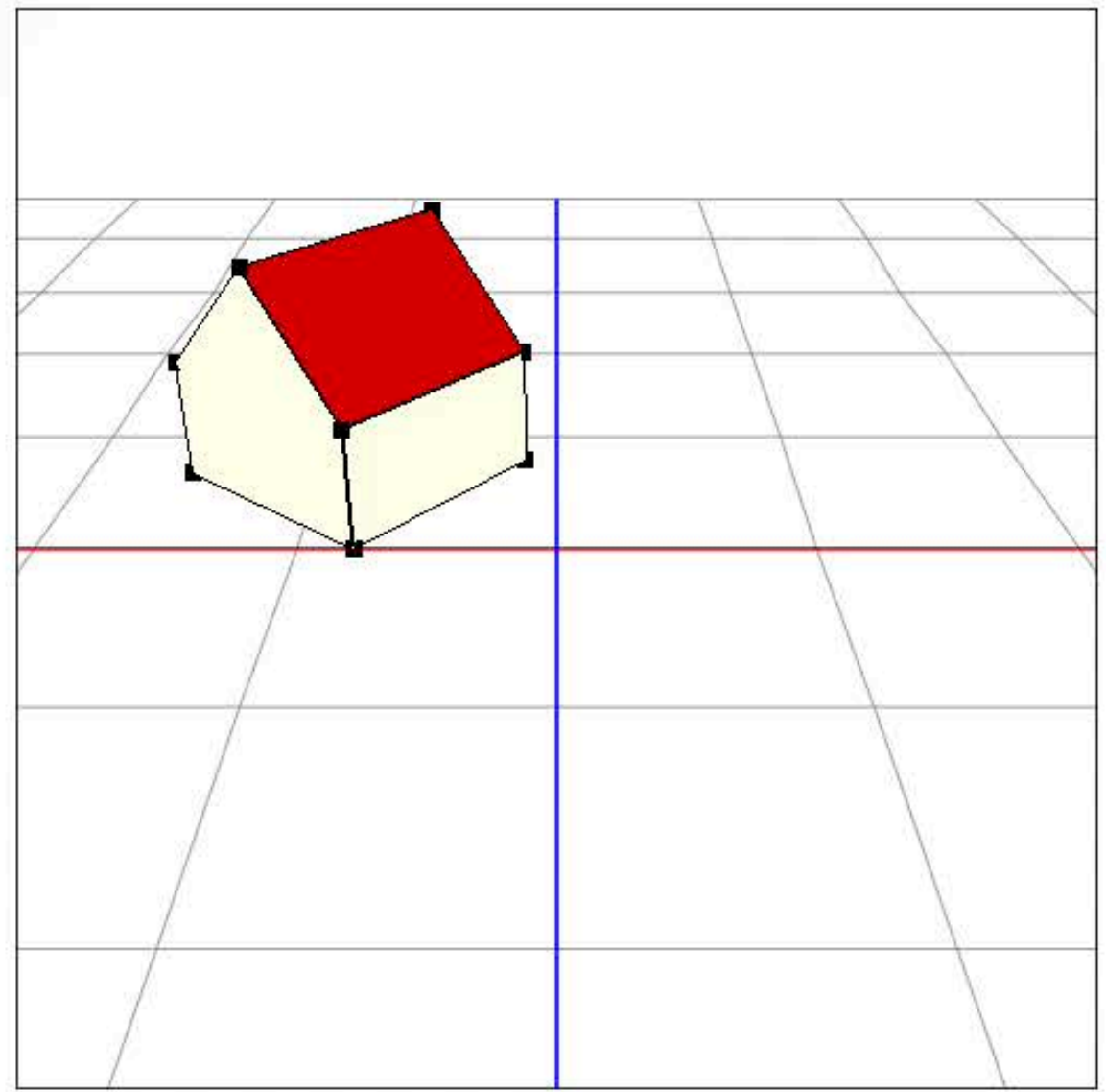


Transformed frame
for link wrt. robot

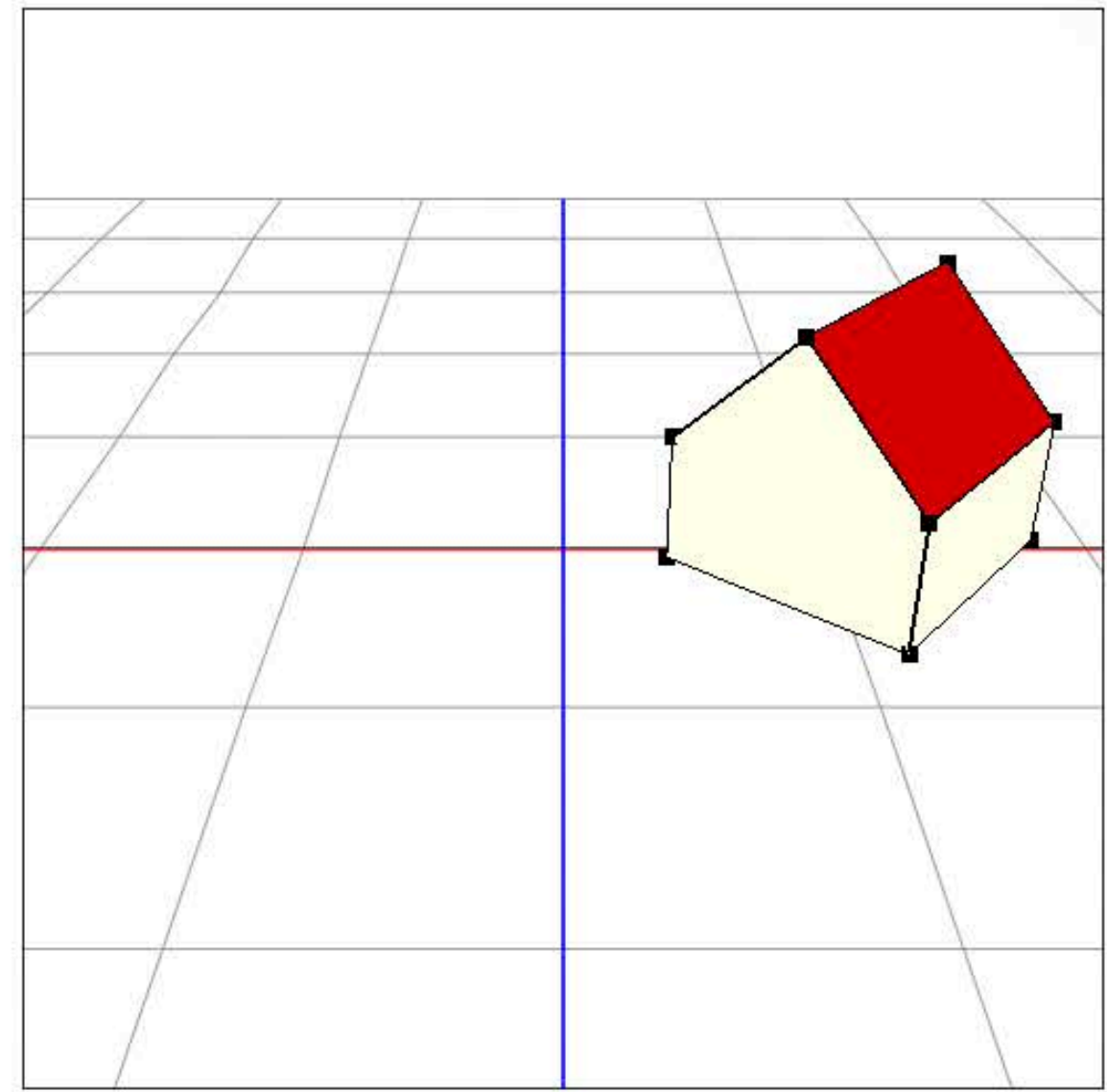


Why not translate then rotate?





$$M = R \cdot T$$



$$M = T \cdot R$$

Note the difference in behavior.

Translation along $x = 1.1$

Rotation about $y = 140^\circ$



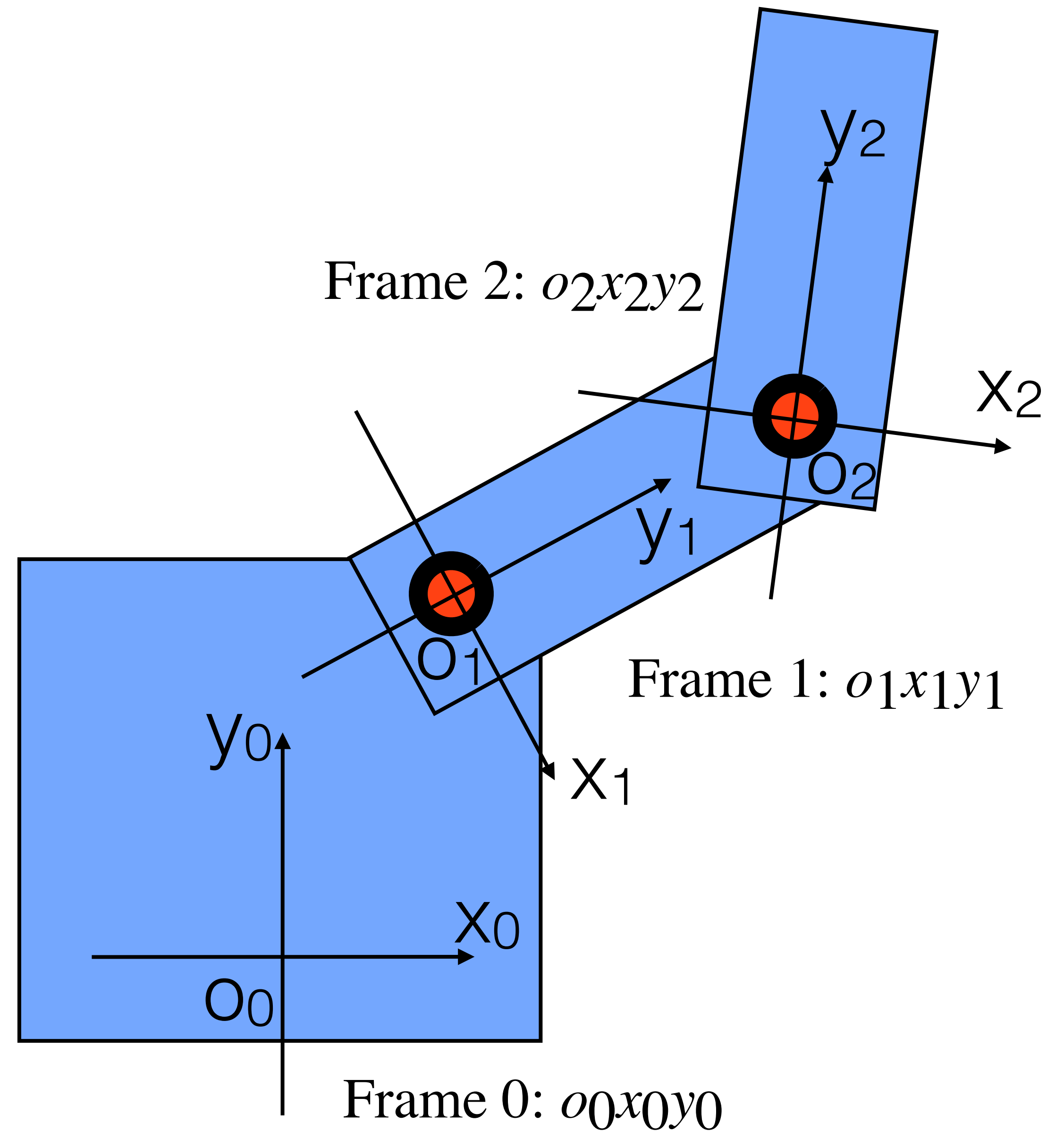
Can we compose multiple frame
transforms?

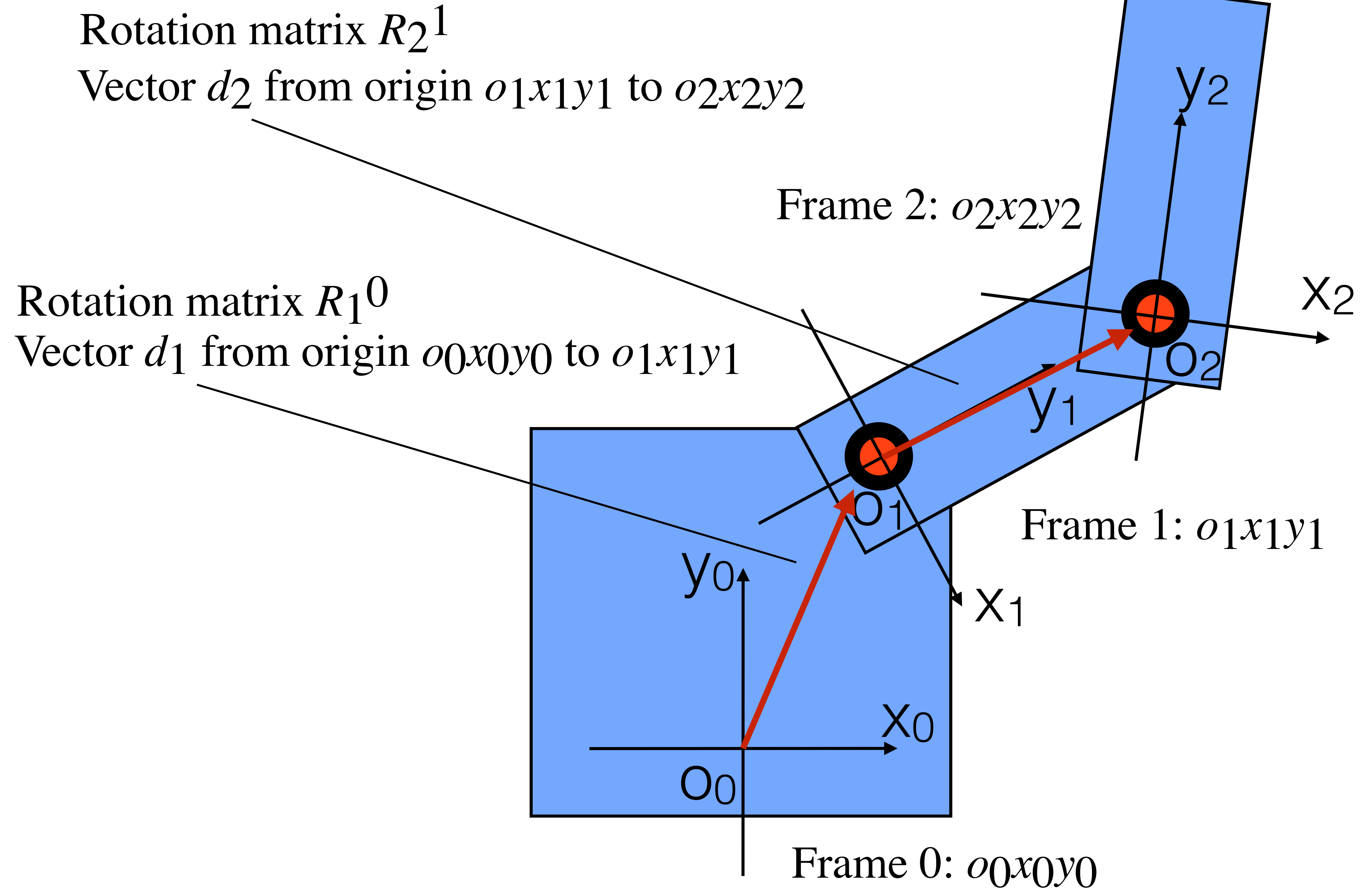


Can we compose multiple frame
transforms?

Consider the 3 frames of a
planar 2-link robot







A point in frame 1 relates to a point in frame 0 by

$$p^0 = R_1^0 p^1 + d_1^0$$

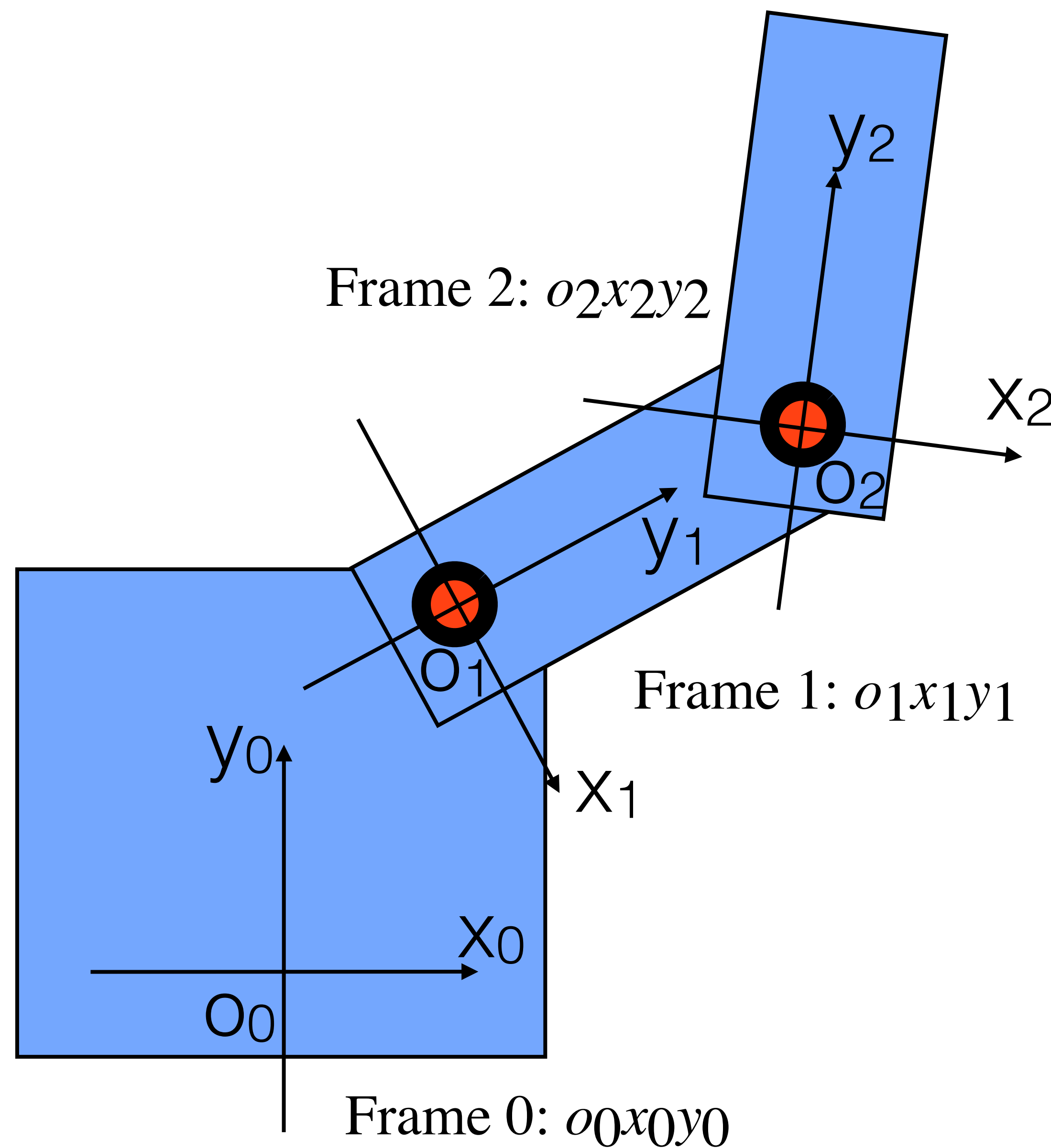
and point in frame 2 relates to point in frame 1 by

$$p^1 = R_2^1 p^2 + d_2^1$$

By substitution of p^1 into the expression for p^0 ,

a point in frame 2 relates to a point in frame 0 by

$$p^0 = \underbrace{R_1^0 R_2^1}_{R_2^0} p^2 + \underbrace{R_1^0 d_2^1 + d_1^0}_{d_2^0}$$



$$\begin{bmatrix} R_1^0 & d_1^0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} R_2^1 & d_2^1 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} R_1^0 R_2^1 & R_1^0 d_2^1 + d_1^0 \\ 0 & 1 \end{bmatrix}$$

Alternatively, relation expressed by composed transform from frame 2 to frame 0 as:

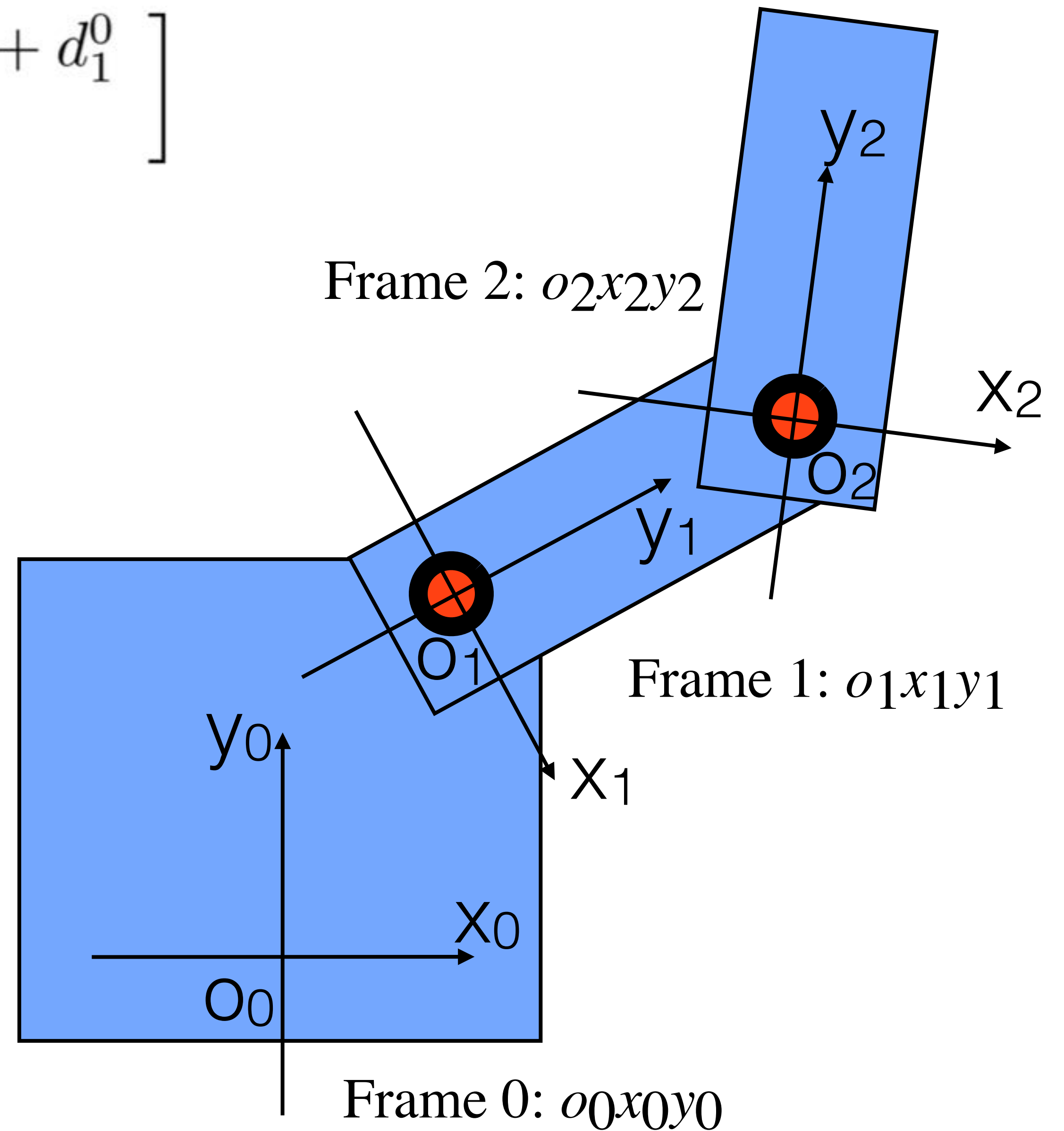
$$p^0 = R_2^0 p^2 + d_2^0$$

where

$$R_2^0 = R_1^0 R_2^1$$

$$d_2^0 = R_1^0 d_2^1 + d_1^0$$

which can be observed by block multiplying transforms



$$\begin{bmatrix} R_1^0 & d_1^0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} R_2^1 & d_2^1 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} R_1^0 R_2^1 & R_1^0 d_2^1 + d_1^0 \\ 0 & 1 \end{bmatrix}$$

Alternatively, relative transform from frame

$$p^0 = R_2^0 p^2 + d_2^0$$

where

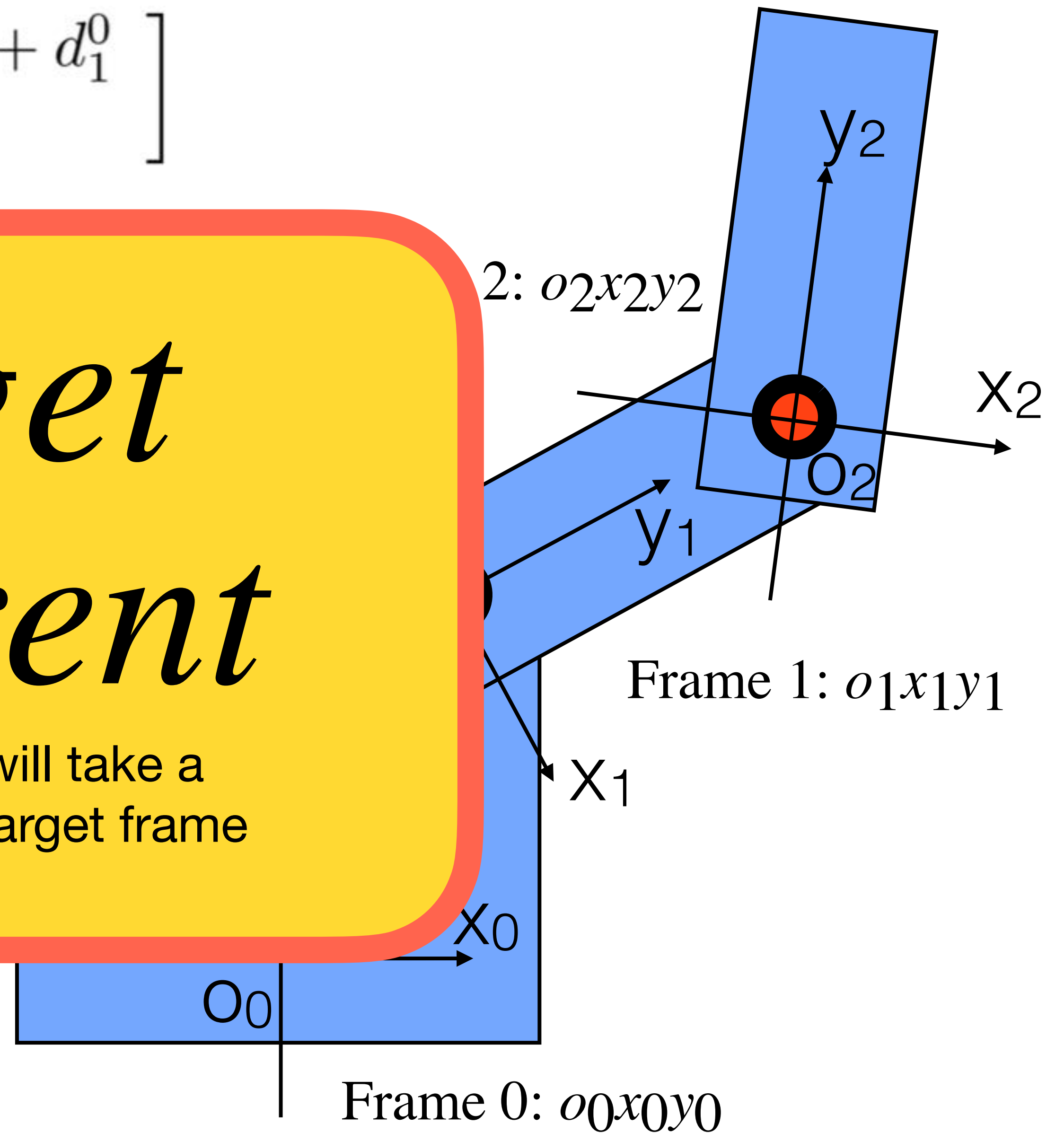
$$R_2^0 = R_1^0 R_2^1$$

$$d_2^0 = R_1^0 d_2^1$$

which can be observed by block multiplying transforms

R *target* *current*

Rotation Matrix that will take a point in current to the target frame



How do we extend this to 3D?



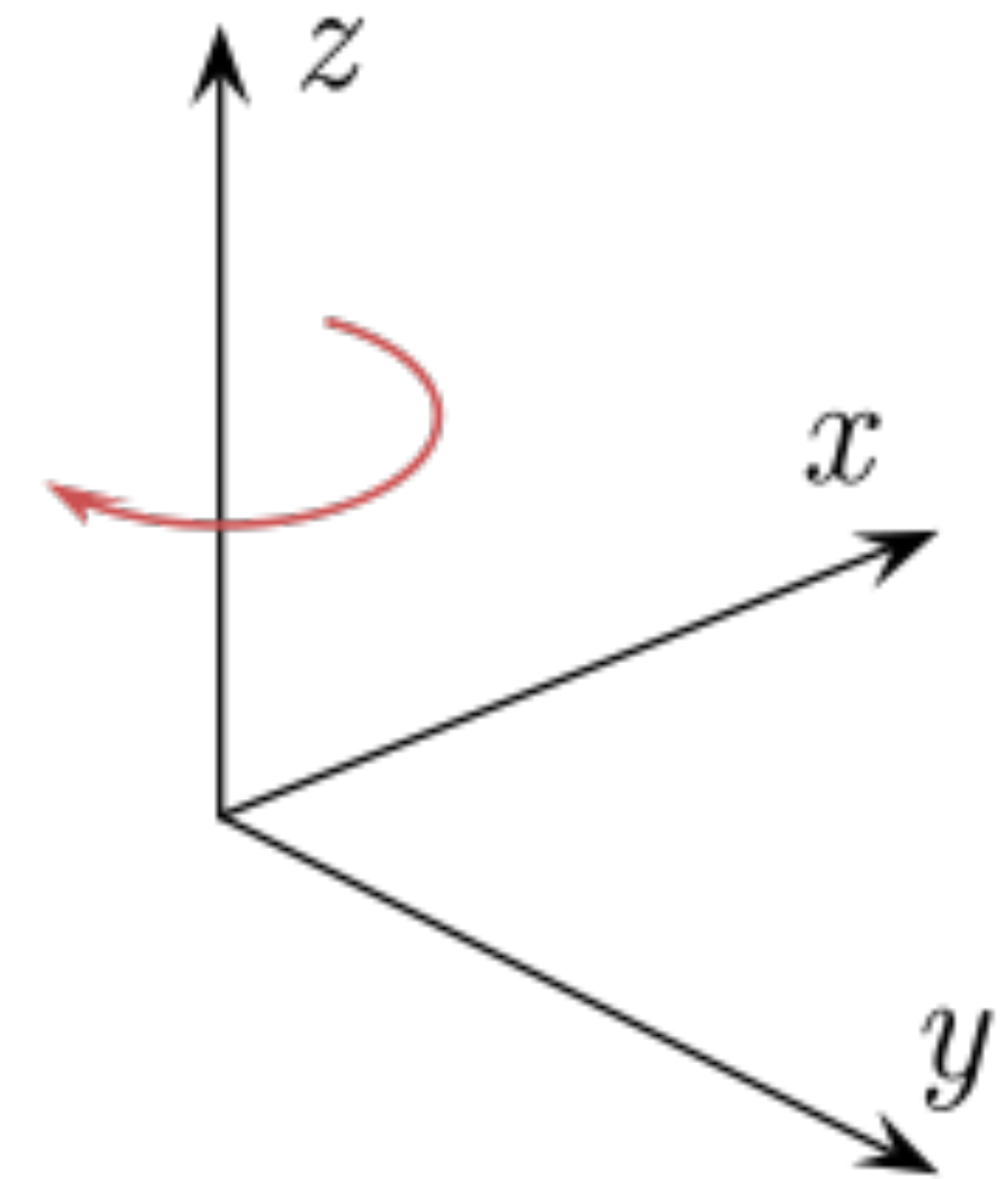
3D Translation and Rotation

$$D(d_x, d_y, d_z) \begin{bmatrix} 1 & 0 & 0 & d_x \\ 0 & 1 & 0 & d_y \\ 0 & 0 & 1 & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$R_z(\theta) \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 & 0 \\ \sin(\theta) & \cos(\theta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

2D rotation in 3D is rotation about Z axis

R_z



3D Translation and Rotation

$$D(d_x, d_y, d_z) \begin{bmatrix} 1 & 0 & 0 & d_x \\ 0 & 1 & 0 & d_y \\ 0 & 0 & 1 & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$R_x(\theta) \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) & 0 \\ 0 & \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$R_z(\theta) \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 & 0 \\ \sin(\theta) & \cos(\theta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$R_y(\theta) \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



3D Homogeneous Transform

Rotate about each axis in order $R = R_x(\Theta_x) R_y(\Theta_y) R_z(\Theta_z)$

$$\begin{bmatrix} 1 & 0 & 0 & d_x \\ 0 & 1 & 0 & d_y \\ 0 & 0 & 1 & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) & 0 \\ 0 & \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 & 0 \\ \sin(\theta) & \cos(\theta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$D(d_x, d_y, d_z)$ $R_x(\theta)$ $R_y(\theta)$ $R_z(\theta)$



3D Homogeneous Transform

$$\begin{bmatrix} 1 & 0 & 0 & d_x \\ 0 & 1 & 0 & d_y \\ 0 & 0 & 1 & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) & 0 \\ 0 & \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 & 0 \\ \sin(\theta) & \cos(\theta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= H_3 = \begin{bmatrix} R_{00} & R_{01} & R_{02} & d_x \\ R_{10} & R_{11} & R_{12} & d_y \\ R_{20} & R_{21} & R_{22} & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}_{3 \times 3} & \mathbf{d}_{3 \times 1} \\ \mathbf{0}_{1 \times 3} & 1 \end{bmatrix}$$

$$\begin{aligned} H_3 &\in SE(3) \\ \mathbf{R}_{3 \times 3} &\in SO(3) \\ \mathbf{d}_{3 \times 1} &\in \mathcal{R}^3 \end{aligned}$$



3D Homogeneous Transform

$$H_3 = \begin{bmatrix} R_{00} & R_{01} & R_{02} & d_x \\ R_{10} & R_{11} & R_{12} & d_y \\ R_{20} & R_{21} & R_{22} & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}_{3 \times 3} & \mathbf{d}_{3 \times 1} \\ \mathbf{0}_{1 \times 3} & 1 \end{bmatrix} \in SE(3)$$

if $T_1^0 \in SE(3)$ and $T_2^1 \in SE(3)$ then composition holds:

$$\begin{bmatrix} R_1^0 & d_1^0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} R_2^1 & d_2^1 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} R_1^0 R_2^1 & R_1^0 d_2^1 + d_1^0 \\ 0 & 1 \end{bmatrix}$$

such that points in Frame 2 can be expressed in Frame 0 by:

$$p^0 = T_1^0 T_2^1 p^2$$



Next lecture:
Representations II:
Rotations & Quaternions





PR2 Fetches Sandwich from Subway 12 years ago!

Autonomous Subway sandwich delivery
by a PR2 robot,
from the University of Tokyo and TUM